NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

PSK SHIFT TIMING INFORMATION DETECTION USING IMAGE PROCESSING AND A MATCHED FILTER

by

Joseph E. Kramer

September 2009

Thesis Advisor:	Monique P. Fargues
Second Reader:	Roberto Cristi

Approved for public release; distribution is unlimited

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY ***(Leave blank)***	**2. REPORT DATE** September 2009	**3. REPORT TYPE AND DATES COVERED** Master's Thesis	
4. TITLE AND SUBTITLE PSK Shift Timing Information Detection Using Image Processing and a Matched Filter		**5. FUNDING NUMBERS**	
6. AUTHOR(S) Kramer, Joseph E.			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000		**8. PERFORMING ORGANIZATION REPORT NUMBER**	
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES)		**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**	
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.		**12b. DISTRIBUTION CODE**	
13. ABSTRACT (maximum 200 words) This thesis investigates the detection of phase shifts contained within a noisy Phase Shift Keyed modulated signal. The approach derived in this work applies image-processing techniques including two-dimensional filters, edge detection, morphological processing, and a two-dimensional cross-correlation matched filter to detect the phase shifts from the angle of the signal temporal correlation function. Results show the proposed algorithm is robust to additive white noise distortions down to SNR levels of 4 dB.			
14. SUBJECT TERMS Phase Shift Keyed signals, Image Processing, Temporal Correlation Function, Edge Detection, Morphological operations, Two-dimensional Matched Filter			**15. NUMBER OF PAGES** 153
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

PSK SHIFT TIMING INFORMATION DETECTION USING IMAGE PROCESSING AND A MATCHED FILTER

Joseph E. Kramer
Lieutenant, United States Navy
B.S.E.E., University of Oklahoma, 2000
M.S.E.M., Old Dominion University, 2007

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 2009

Author: Joseph E. Kramer

Approved by: Monique P. Fargues
Thesis Advisor

Roberto Cristi
Second Reader

Jeffrey B. Knorr
Chairman
Department of Electrical and Computer Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

This thesis investigates the detection of phase shifts contained within a noisy Phase Shift Keyed modulated signal. The approach derived in this work applies image-processing techniques including two-dimensional filters, edge detection, morphological processing, and a two-dimensional cross-correlation matched filter to detect the phase shifts from the angle of the signal temporal correlation function. Results show the proposed algorithm is robust to additive white noise distortions down to SNR levels of 4 dB.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

EXECUTIVE SUMMARY

The United States is currently conducting counterinsurgency operations against asymmetric combatants in Middle Eastern regions and worldwide. These combatants use unconventional tactics to execute deadly attacks on military and civilian targets coordinated through normal communications in a covert manner. The techniques this research proposes will prove beneficial in decoding intercepted communications of possible pre-emptive hostile actions. This matched filter approach to tracking phase shifts contained within Quadrature-Phase Shift Keyed (QPSK) signals provides a necessary capability to dissect PSK signals so decoding schemes can be employed.

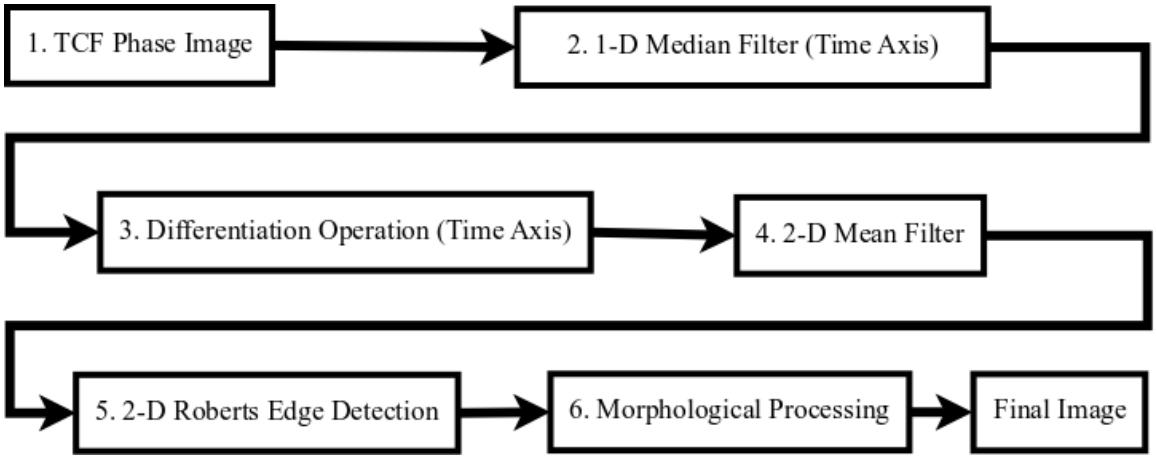
PSK signal encoding is applied in a number of applications in today's communications industry, including VHF and UHF television, FM radio, terrestrial and satellite microwave applications as well as short point-to-point experimental applications [1]. Wireless LAN as well as satellite communications require disciplined bandwidth conservation and make extensive use of Frequency Shift Keyed (FSK), Amplitude Shift Keyed (ASK), and QPSK types of compact signal generation and transmission schemes. The QPSK modulation technique was investigated because its bit encoding efficiency is commonly used to support modern communication requirements.

The Temporal Correlation Function (TCF) is a two-dimensional instantaneous complex symmetric auto-correlation function used to find the phase shift characteristic on a per-data frame basis. The objectives of this research were the following:

- Investigate an approach to identify phase shift timing within QPSK signals using image processing techniques and the Temporal Correlation Function (TCF) phase computation.
- Develop and test the image processing techniques required to remove noise from TCF phase images such that phase shifts are enhanced.
- Develop, design, and test the resulting phase shift identification scheme.

- Develop, design, and test an optional analysis window overlapping technique to improve phase shift detection rates in noisy environments.

The designed phase shift time detection scheme is split into two main phases. The first phase computes the initial TCF phase matrix and transforms it through a series of pre-processing filters and morphological operations into an image better suited for a matched filter solution. Second, the actual phase shift timing extraction task is conducted using a two-dimensional matched filter. Pre-processing phase steps used are summarized in the following flow chart.



Start to Finish Image Pre-Processing Flow (Reproduction of Figure 8)

The overall phase shift time extraction phase has two main components:

- A two-dimensional cross correlation matched filter operation, and
- An analysis window overlapping option designed to combine decisions obtained on separate windows and to improve performances in noisy conditions.

First, target images and structuring elements are defined followed by the TCF phase computation for each signal segment. These TCF phase images are then processed to isolate the phase shift timing before the two-dimensional cross-correlation locates the

shift time in each window. Last, final phase shift time decisions are derived, either from individual decisions, or by combining multiple decisions when overlapping analysis is allowed during processing.

User-specified parameters are derived for QPSK signals in 6dB SNR environments and the resulting phase shift identification algorithm is investigated for SNR levels in the range -2dB to 12 dB. Detection performances are derived for 500 non-overlapping windows per experiment, where 250 windows include shift and no-shift cases, respectively, spread semi-randomly throughout the length of the signal when window overlapping is selected. The no-overlap window case did not use random placement to ensure only one shift or no shift per window. Results show detection performances are perfect for SNR levels 6 dB and above, and 95% detection rate are obtained between 4 and 6 dB SNR levels. Specific findings are listed below:

- Results obtained without overlapping analysis windows showed 99.2% accuracy rates at 6 dB.
- 60% and 90% window overlapping results showed detection performances are equal to 100% at the 10% tolerance level for 6 dB SNR level.
- Results showed the differentiation step implemented using the basic difference filter led to better detection performances than using the Savitzky-Golay (SG) FIR filter because the algorithm was designed for the difference filter. SG filters proved to obtain higher detection rates during high noise runs, however, false alarm rates caused by mismatched structuring elements during noise removal and phase shift transformation resulted in lower accuracy rates than the difference filter test results.
- Low and high normalized signal carrier frequencies were shown to have no direct impact on detection performances.

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

First, Deo gratias.

Second, to my beautiful wife, Teresa, and our wonderful children, without whose love and support I would not have had the strength to complete this task.

Finally, to my thesis advisor and instructor, professor Fargues, whose experience and tireless efforts made this thesis one of the most rewarding assignments in my life.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. BACKGROUND

Phase Shift Keyed (PSK) signals form an important part of our private and public communications infrastructure due to the bandwidth utilization benefits offered by PSK modulation schemes. Increased bandwidth efficiency is not lost on technologies such as wireless LAN standards where faster data rates are desired or applications where quadrature-PSK and at times eight- or sixteen-PSK are used for increased efficiency. Recall that PSK, similar to Frequency Shift Keyed (FSK), Amplitude Shift Keyed (ASK), and other modulation schemes, is a modulation technique where binary data is encoded from a digital format into an analog format for transmission in a communication system [2].

B. OBJECTIVE

The objectives of this research were the following:

- Investigate an approach to identify phase shift timing within Quadrature Phase Shift Keyed (QPSK) signals using image processing techniques and the Temporal Correlation Function (TCF) phase computation.
- Develop and test the image processing techniques required to remove noise from TCF phase images such that phase shifts are enhanced.
- Develop, design, and test the resulting phase shift identification scheme.
- Develop, design, and test an optional analysis window overlapping technique to improve phase shift detection rates in noisy environments.

C. RELATED WORK

Previous studies identified hop timing information for FSK modulated signals [3], [4]. This research focuses on shift timing identification for PSK modulated signals. However, both studies use the two-dimensional Temporal Correlation Function (TCF) as

the initial transformation step from which shift timing information is to be extracted. Overdyk's research focused on one-dimensional approaches using the discrete wavelet transform to track signal discontinuities within the TCF output [3]. Cheng capitalized on the two-dimensional aspects of the TCF by using morphological image processing techniques to find image edges that enabled him to identify FSK hop times with some success [4]. As a continuation of these ideas, this thesis explores a more advanced textural analysis of the TCF output, and resets the shift timing identification task as a pattern matching problem by using a 2-dimensional matched filter approach to extract shift times present in PSK signals.

D. APPROACH

The designed phase shift time detection scheme is split into two main phases. The first phase computes the initial TCF phase matrix and transforms it through a series of pre-processing filters and morphological operations into an image better suited for a matched filter solution. Second, the actual phase shift timing extraction task is conducted using a two-dimensional matched filter.

The overall phase shift time extraction phase has two main components:

- A two-dimensional cross correlation matched filter operation, and
- An analysis window overlapping option designed to combine decisions obtained on separate windows and to improve performances in noisy conditions.

Overall, this thesis is composed of seven chapters including the introduction. Chapter II introduces the underlying encoding scheme used in PSK signals to understand the intent of this research from a user perspective. Chapter III develops the TCF function obtained from PSK signals. Chapter IV presents the image processing techniques used to mitigate noise effects and improve the detection performance of the final algorithm. Chapter V details the actual MATLAB/SIMULINK model and algorithm, the threshold analysis and decision process. Finally, results and conclusions are summarized in Chapters VI and VII.

II. PSK COMMUNICATION SIGNALS

This section introduces the basic concepts used during this research to simulate Quadrature Phase Shift Keyed (QPSK) signals.

A. SIMULATION SIGNAL

Phase Shift Keyed (PSK) signal encoding applies to a number of applications in today's communications industry. PSK applications include VHF and UHF television, FM radio, terrestrial and satellite microwave applications as well as short point-to-point experimental applications [1]. Wireless LAN as well as satellite communications require disciplined bandwidth conservation and make extensive use of Frequency Shift Keyed (FSK), Amplitude Shift Keyed (ASK), and PSK types of compact signal generation and transmission schemes. The basic Binary Phase Shift Keyed (BPSK) signal involves encoding data using a carrier frequency by changing the signal phase between 0° or 180° [2]. Furthermore, QPSK schemes allow the user to double data rates for the same bandwidth consumption or halve the bandwidth consumption while still transmitting the same data rate by more efficient bit encoding.

QPSK uses phase differences of 90° to divide the state space into four quadrants. Higher M-ary Phase Shift Keying (MPSK) schemes can be defined to conserve more bandwidth [2]. QPSK was chosen as the initial test signal for this analysis because of its simple yet real world representation of a range of MPSK signals. Furthermore, the QPSK symbol configuration shown below was chosen as a real world representation of actual PSK phase shifts for a baseline analysis [2]. The signal model used in this work takes on the following mathematical form:

$$x_{QPSK}(t) = A\cos(2\pi f_c t + \tilde{\theta}_M), \tag{2.1}$$

where A is the amplitude and for QPSK encoding we have,

$$\tilde{\theta}_M = \frac{\pi}{4} + \frac{\pi}{2} \bullet M, \quad M = \{0,1,2,3\}. \tag{2.2}$$

Equations (2.1) and (2.2) combine to form the analytical QPSK signal where $\tilde{\theta}_M$ is restricted to four phase values with a state space as illustrated in Figure 1.

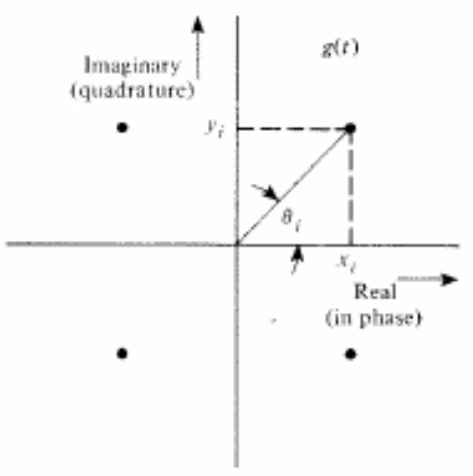


Figure 1. QPSK constellation diagram, from [2, p. 320].

B. SIGNAL ASSUMPTIONS

The goal of this algorithm is to extract phase shift timing information from QPSK modulated signals. The mathematical description discussed in the first part of this chapter is the accepted implementation of a QPSK signal, although, in hardware it is easier to use two out of phase sine waves to generate the appropriate quadrature phase result based on the bit stream being sent [2]. Three signal assumptions are made in this work:

- First, we assume the generated signal is sampled high enough to avoid aliasing.
- Second, we assume a starting phase offset for the signal. Actual phase offsets follow the derivation described in Section II.A above from [2]. Note that the TCF output discussed in the next chapter shows that the quality of the phase shift detection step depends on the difference between successive signal phase values not on their individual values.

- Third, the minimum bit period although not specifically defined is one assumption that will affect the results. The signal is constructed so that phase shifts are semi-randomly distributed between the beginning of the second window frame of the signal and the second to last window frame of the signal to avoid issues dealing with first and last signal window frames. However, the signal can be initialized so that some phase shifts are very close, resulting in two phase shifts in a given analysis frame. The algorithm is intended to address this potential problem by combining consecutive decisions obtained over overlapping windows.

The next section discusses the temporal correlation function and its place in the identification algorithm before discussing the details of the algorithm itself.

THIS PAGE INTENTIONALLY LEFT BLANK

III. TEMPORAL CORRELATION FUNCTION (TCF)

Equation Chapter 3 Section 1This section introduces the Temporal Correlation Function (TCF) obtained from analytic Quadrature Phase Shift Keyed (QPSK) and Frequency Shift Keyed (FSK) signals. It also describes and discusses the differences observed between the TCF functions generated from both modulating schemes.

A. BASIC DESCRIPTION

The Temporal Correlation Function (TCF), $TCF_x(t,\tau)$, is the two-dimensional instantaneous auto-correlation function of the signal x(t) defined as:

$$TCF_x(t,\tau) = x\left(t+\frac{\tau}{2}\right)x^*\left(t-\frac{\tau}{2}\right), \tag{3.1}$$

where t represents time and τ is the positive lag value for the computation. Note that it is not necessary to calculate negative lag values of the TCF as it is a complex symmetric function. Additionally, the TCF function is a two-dimensional function which is suitable for image manipulation tools. The TCF was shown to be a useful tool to estimate phase changes as the sections below will demonstrate.

1. FSK Based TCF

In earlier work, Cheng and Overdyk used the TCF expression for Frequency Shift Keyed (FSK) signals as the basic tool to detect FSK hopping times [3, 4]. A basic analytic FSK signal with frequency shift at time T_{shift} is defined around this shift instant as:

$$x_{FSK}(t) = e^{i(2\pi f_1 t+\theta)}\left[u(t)-u(t-T_{shift})\right]+e^{i(2\pi f_2 t+\theta)}\left[u(t-T_{shift}+1)-u(t-T)\right], \tag{3.2}$$

where f_1 and f_2 are frequencies representing two different encoded bits in the FSK scheme and the phase jump, θ, is assumed constant. The analytic FSK expression is derived by taking the Hilbert transform of the real valued FSK signal expression. Equation (3.2) uses unit step functions to define the starting points of the respective frequency encoded

bits of the FSK signal. Thus, the unit step functions define the beginning of the signal with frequency, f_1, at some arbitrary time t, the time of hop from frequency f_1 to frequency f_2 at time T_{shift} and the end of the signal with frequency f_2 at time T.

Applying the signal in Equation (3.2) to Equation (3.1) yields the following TCF expression:

$$TCF_{FSK}(t,\tau) = \begin{cases} e^{i(2\pi f_1 \tau)}\left[u(t-\frac{\tau}{2}) - u(t+\frac{\tau}{2}-T_{shift})\right] \\ +e^{i2\pi\left[(f_2-f_1)t+\left(\frac{f_1+f_2}{2}\right)\tau\right]}\left[u(t+\frac{\tau}{2}-T_{shift}+1) - u(t-\frac{\tau}{2}-T_{shift})\right] \\ +e^{i(2\pi f_2 \tau)}\left[u(t-\frac{\tau}{2}-T_{shift}+1) - u(t+\frac{\tau}{2}-T)\right] \end{cases} \quad (3.3)$$

$$= \quad TCF_1(t,\tau) + TCF_{12}(t,\tau) + TCF_2(t,\tau),$$

where t and τ are referred to as the time and lag indices, respectively. Note that the FSK-based TCF expression shown in Equation (3.3) has a very similar analytic structure to the PSK-based version upon which this research focuses. The TCF function derived for an FSK signal has three non-overlapping regions called TCF_1, TCF_2, and TCF_{12}. Figure 2 presents the phase of the TCF expression obtained for such an FSK signal with f_1=0.1 Hz and f_2=0.3 Hz, normalized frequencies respectively, where the center region, TCF_{12}, is highlighted by the dashed box. The two-dimensional TCF phase function output is shown with the time parameter on the vertical axis and the lag parameter on the horizontal axis allowing us to identify phase shift instants. It turns out that the phase shift time is located at the intersection of the three region boundaries along the time axis.

Additive white Gaussian noise (AWGN) inserted into the original real valued FSK signal prior to the analytic transformation and the TCF computation changes the cross-term region considerably, as seen in Figure 2.b.

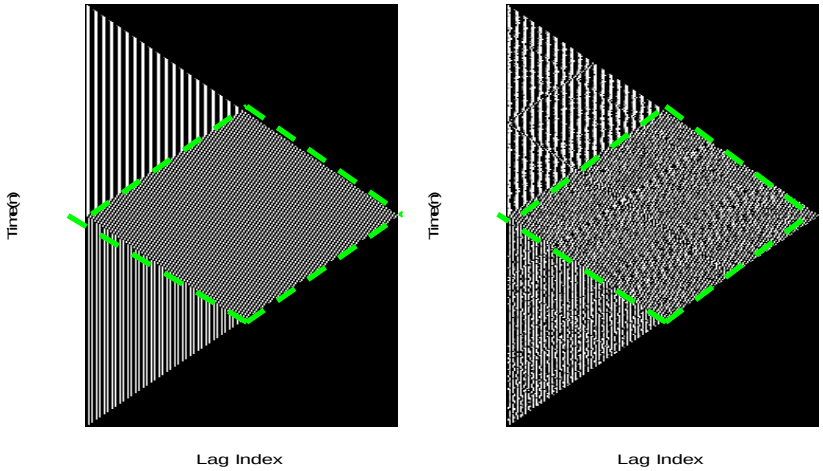


Figure 2. TCF output following phase computation; a. FSK, Noise free, and b. FSK + AWGN, SNR = 6 dB.

2. PSK Based TCF

The analytic PSK signal, $x_{PSK}(t)$, defined around the phase shift instant T_{shift} is defined as:

$$x_{PSK}(t) = e^{i(2\pi ft+\theta_1)}\left[u(t)-u(t-T_{shift})\right] + e^{i(2\pi ft+\theta_2)}\left[u(t-T_{shift}+1)-u(t-T)\right], \qquad (3.4)$$

where the frequency, f, of the signal is assumed constant and the components before and after the phase shift instant T_{shift} have phases equal to θ_1 and θ_2, respectively, where θ_1 and θ_2 are assumed to be different. Similar to the FSK case described above, unit step functions define the beginning of the signal with phase, θ_1, at some arbitrary time, t, the time of shift from phase, θ_1, to phase, θ_2, at time, T_{shift}, and the end of the signal with phase, θ_2, at time, T. Furthermore, Equation (3.4) represents one unique phase shift for a generic Mary-Phase Shift Keyed (MPSK) signal, where the actual shift value is constrained by the specific MPSK scheme investigated. In this work, results are examined on a per-shift basis and PSK and QPSK terms are used interchangeably throughout this section when discussing the QPSK signal used in this research.

The PSK signal based TCF output is derived by substituting the PSK signal expression in Equation (3.4) into Equation (3.1). The PSK signal derived TCF phase expression looks similar to that obtained using a FSK signal with the exception that cross-terms are now composed of a single phase difference instead of sums and differences of frequencies [3]. Thus, the TCF expression obtained for a PSK signal, $x_{PSK}(t)$, is given as follows:

$$TCF_x(t,\tau) = \begin{cases} e^{i(2\pi f\tau)}\left[u(t-\frac{\tau}{2}) - u(t+\frac{\tau}{2}-T_{shift})\right] \\ +e^{i(2\pi f\tau+(\theta_2-\theta_1)}\left[u(t+\frac{\tau}{2}-T_{shift}+1) - u(t-\frac{\tau}{2}-T_{shift})\right] \\ +e^{i(2\pi f\tau)}\left[u(t-\frac{\tau}{2}-T_{shift}+1) - u(t+\frac{\tau}{2}-T)\right] \end{cases} \quad (3.5)$$

$$= \quad TCF_1(t,\tau) + TCF_{12}(t,\tau) + TCF_2(t,\tau),$$

where $TCF_1(t,\tau)$, $TCF_{12}(t,\tau)$, and $TCF_2(t,\tau)$ represent the three non-overlapping regions included in Equation (3.5), respectively. Note that individual regions "one" and "two" covered by TCF_1 and TCF_2 are identical for the PSK case, as phase differences exhibited in the FSK case cancel out when $f_1 = f_2 = f$. Thus, there is no difference in frequency along the time and lag axes in these two regions. As a result, the PSK signal derived TCF phase function does not exhibit a change in texture characteristics from region to region as was the case for the FSK based TCF phase function.

Figure 3 plots the phase of a PSK signal derived TCF function. Note the lack of texture change between the three TCF regions for the PSK-based case. The one exploitable quality for the PSK signal case is the fact that the center region, $TCF_{12}(t,\tau)$, exhibits a shift along the lag axis, forming discontinuities that appear as rough boundary edges around that center region. However, the actual phase shift is only slightly visible in a pure analytic signal, as shown in Figure 3. The phase shift information becomes almost invisible when the PSK signal is distorted by AWGN of SNR=6 dB because noise artifacts generate edge features similar to phase shift characteristics, as shown in Figure 3.b. Thus, image-processing tools need to be applied to clean up noise distortions and render a more visible phase shift time.

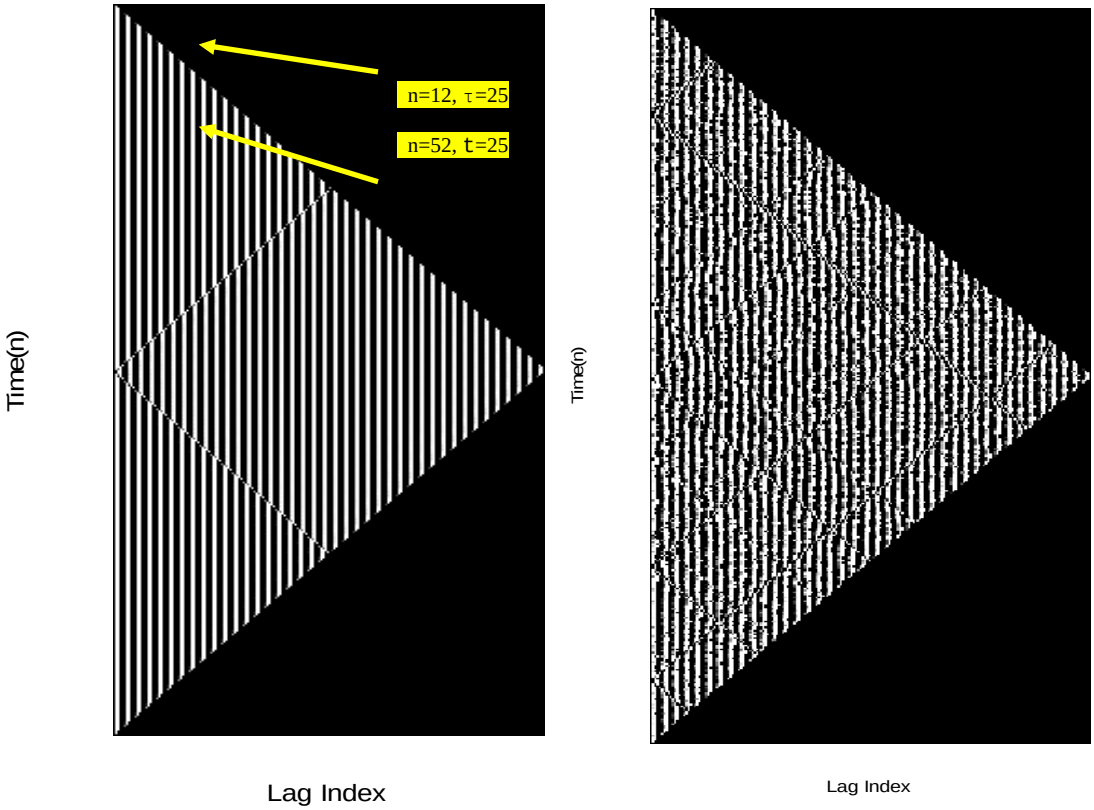


Figure 3. TCF phase output; a. QPSK No Noise, and b. QPSK + AWGN, SNR = 6 dB.

Figures 4 and 5 plot the expressions for the TCF phase output obtained for clean and noisy QPSK signals at a fixed lag τ=25 obtained from the TCF phase function shown in Figure 3. Note the zero time index starts at the top left corner of each image. These figures show that the output is equal to zero for time index n<25 due to the triangular shape of the TCF function as described above. Two points are shown in Figure 3 illustrating this idea. Figures 4 and 5 also show that the phase shift time information contained in the edges of the $TCF_{12}(t,\tau)$, located at n=125 and n=175 for a simulated phase shift at n=150, become noisier when the signal is noisy, but is still detectable.

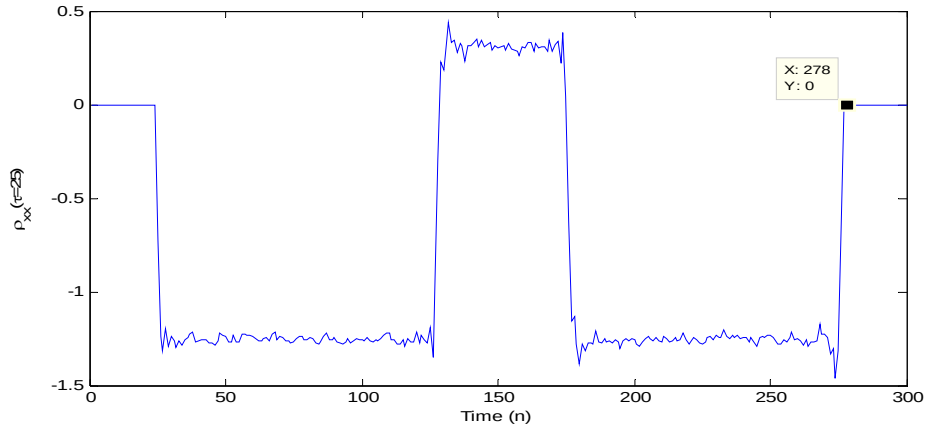


Figure 4. TCF phase output for τ=25 in clean QPSK case.

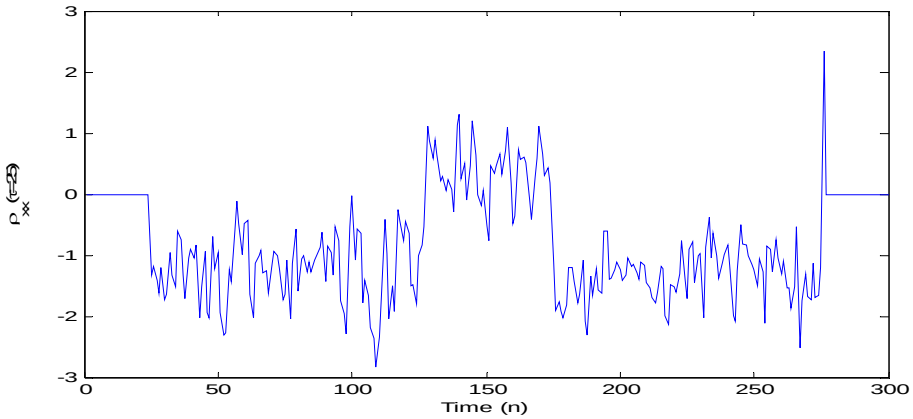


Figure 5. QPSK + AWGN, SNR=6 dB, TCF phase output, τ=25.

3. FSK and PSK Based TCF Comparisons

Note that the change from frequency shift to phase shift encoding makes the TCF-based phase shift time detection scheme less robust to noise distortions. As mentioned earlier, the TCF middle region, denoted by TCF_{12} in Equation (3.3), shows frequency terms along time and lag axes for FSK signals which are different from those observed in TCF_1 and TCF_2 regions. However, this characteristic does not exist for PSK signals, where the same frequency terms are observed along time and lag axes in these two regions, and the difference with region TCF_{12} comes from the offset at the boundaries.

As a result, the phase shift time information is contained in the "misalignment" between the vertical lines shown in Figure 6, and the PSK phase shift time detection task is likely to be more difficult.

Figure 6 compares clean FSK and QPSK TCF phase outputs to illustrate this textural difference. As noted above, phase shift time information is extracted from the transition edges between the three TCF phase regions. However, noise distortions have a significant effect on the results obtained as they make these edges harder to identify and extract.

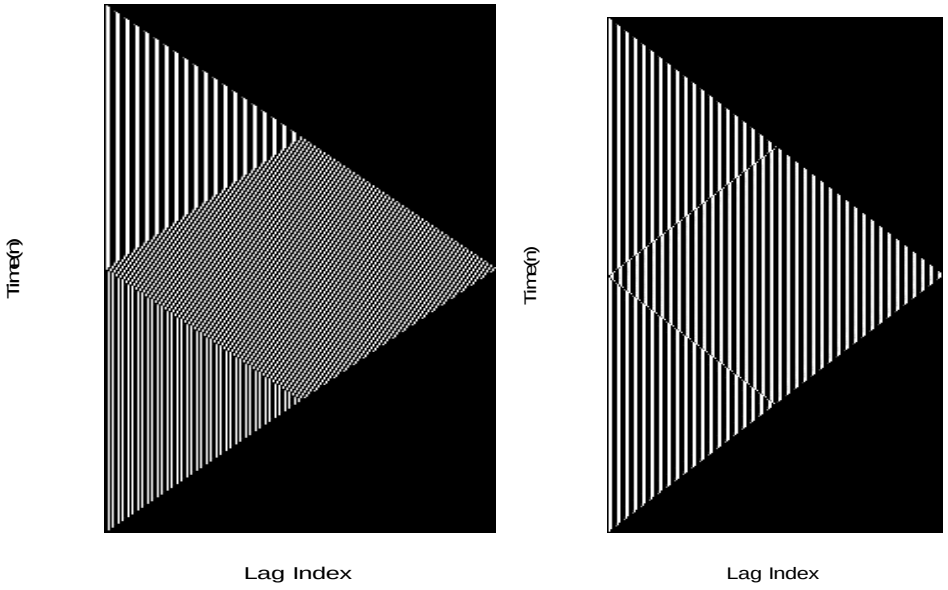


Figure 6. TCF phase output, noise-free signal cases: a. FSK hop signal, b. QPSK phase shift signal.

B. OVERALL PHASE SHIFT DETECTION

This section discusses the role the TCF phase function plays in detecting phase shift times. The overall phase shift time detection scheme is split into two main phases. The first phase computes the initial TCF phase matrix and transforms it through a series of filters and morphological operations into an image better suited for the second phase, where the actual phase shift timing extraction task is conducted. As mentioned earlier, edges around the center region of the TCF phase output function contain phase shift time

information, and extracting such edge information is the primary focus of this work. Figure 7 shows an original TCF phase image for a typical noisy phase shift image (SNR=6 dB) and the final image obtained after processing to be used for phase shift time detection.

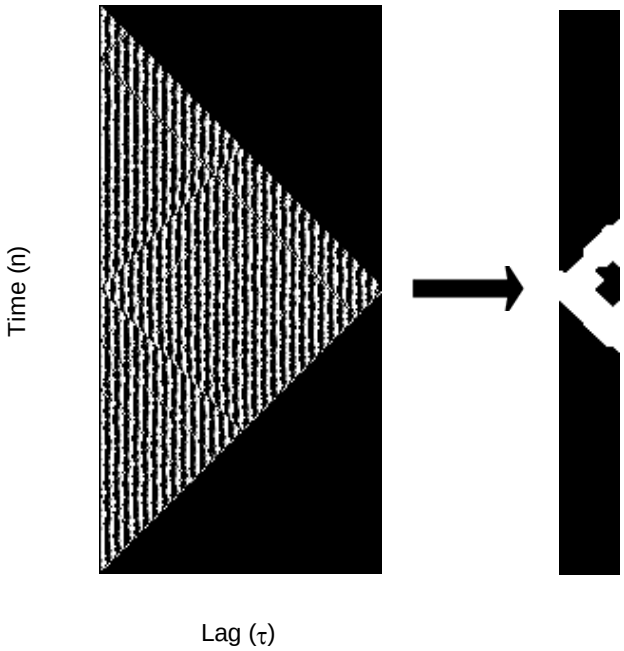


Figure 7. TCF phase output images: Beginning + AWGN, SNR = 6 dB, and Final.

Figure 8 gives a brief overview of the various steps used to transform the initial TCF phase matrix into the final post-processed matrix used to detect the phase shift timing information. Specific details regarding each step are presented later in Chapter V.

- First, the TCF phase image is obtained using the TCF expressions discussed in this chapter.
- Second, a 15^{th} order one-dimensional median filter applied along the time axis is applied to remove noise.
- Third, a 2^{nd} order difference operation or 1^{st} order derivative filter is applied along the time axis to remove most image features leaving only the desired phase shift discontinuities.

- Fourth, a two-dimensional mean filter is applied to remove excess image noise prior to the edge detection stage.
- Fifth, the Roberts edge detection operation is applied to enhance image edges to emphasize phase shift discontinuities.
- Sixth, morphological operations are applied to further enhance the desired edges and remove unwanted noise remnants. At that point, the resulting image is ready for phase shift time extraction.

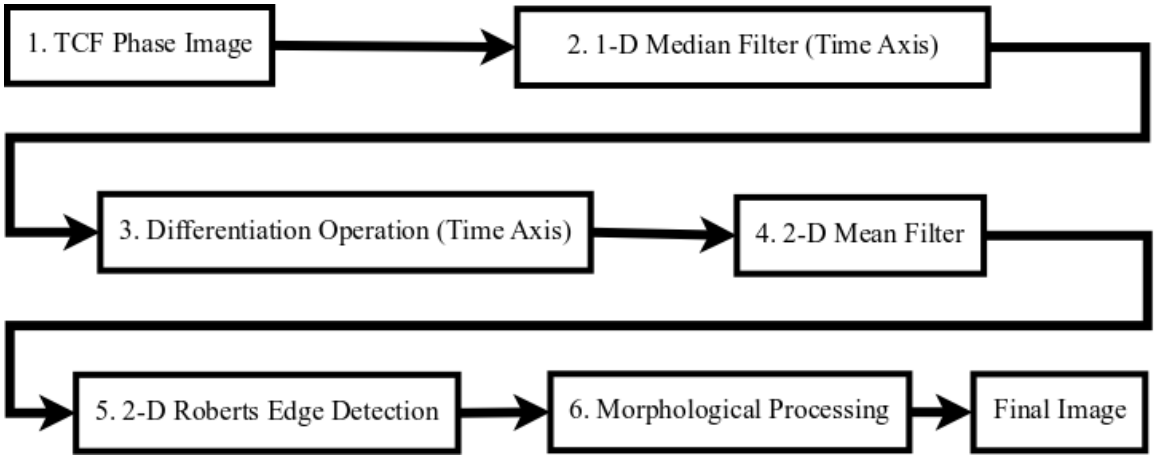


Figure 8. Start to Finish Image Processing Flow.

This section first introduced the basic TCF phase function and its resulting expression for FSK and PSK signals. Next, it presented an overview of the various steps followed to reduce noise impacts and generate the image used for phase shift timing detection. Specific details regarding each processing step followed in the image processing flow are presented next in Chapters IV and V. Chapter IV focuses on the mechanics of each step and its performance within a noisy environment. Chapter V discusses the algorithm implementation of these steps.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. PRE-PROCESSING TECHNIQUES

The primary goal of the pre-processing phase is to reduce the effect of noise distortions present in the received communication signal prior to phase shift time estimation. This research considers a two-dimensional matched filter approach, where the TCF phase image derived from a noisy signal is matched against that of an expected signal image to extract the phase shift time information. First, one-dimensional filters are applied to enhance the TCF phase image and enhance the phase shift discontinuity from the background noise. Next, edge detection and morphological operations are applied to reduce image noise distortions, as discussed below. Note, several commonly used methods designed to remove salt and pepper noise or to smooth or sharpen images were also considered. However, results showed that they did not improve performance.

A. NOISE REDUCTION: ONE-DIMENSIONAL FILTERS

Note that the signal was not filtered prior to computing the TCF phase expression because of the loss of information resulting from that step. First, a one-dimensional filter was applied to the TCF phase image in an effort to maintain as much of the original signal characteristics as possible due to the statistical sensitivity of a phase shift.

1. Median Filter

The one-dimensional median filter is well suited to remove impulsive noise without destroying long-term trends, and is applied to the TCF phase output along the time axis to reduce noise impacts. Simulations showed that applying this filter before differentiating accented discontinuities present in the TCF phase image that contain the phase shift time information. The one-dimensional mean filter was also investigated but not used as it blurred the image discontinuities containing the phase shift timing information.

Simulations showed that image noise needs to be decreased prior to applying the differential filter operation, as the difference operation further enhances noise distortions.

One-dimensional median and mean filters were applied along the time axis and their results compared. Figure 9 shows the TCF phase image produced from a PSK signal with 6dB AWGN filtered along the time axis using 15^{th} order one-dimensional median and mean filters, respectively. Note the phase shift time for all examples is at time sample 150, located in the center of the time axis, with a window frame of length 300 samples. Thus, edges should appear in the phase image around that time value in Figure 9. Both filters remove noise; however, Figure 9 shows that edge information is better preserved with the median filter than it is with the mean filter. Thus, the one-dimensional median filter was used in the final algorithm.

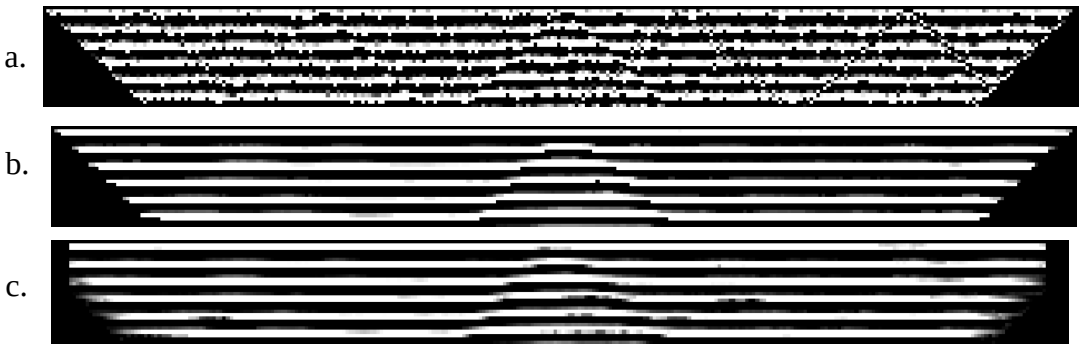


Figure 9. One-dimensional comparison: a. PSK + AWGN, SNR=6dB signal image, b. 15^{th} order Median filter applied along the time axis, c. 15^{th} order Mean filter applied along the time axis (Horizontal axis: Time axis – left to right, Vertical axis: Lag axis – top to bottom).

2. Approximate Derivative Filter

The first filter considered for the derivative operation was the discrete derivative used in the MATLAB function *diff* where differences of neighboring pixels represent the differentiation. The derivative operation is a fast and simple way to accent the discontinuities discussed in the previous chapter that characterize the phase shift time present in the PSK signal. However, the columns of the image must first be filtered to decrease noise distortions and prevent the differentiation step from smearing the image or erasing the desired discontinuity characteristics altogether.

3. Savitzky-Golay Smoothing Filter

The second derivative filter considered was the Savitzky-Golay (SG) smoothing filter for a more noise robust solution to the differentiation operation. The SG filter is a one-dimensional k^{th} order Finite Impulse Response (FIR) filter with coefficients developed based on the minimum least squares error polynomial fit within each data frame [5]. Simulations showed the 1^{st} order derivative SG filter was better than the 0^{th} or 2^{nd} order filters at adequately removing noise while differentiating the image. Specifically, the 2^{nd} order SG filter erased phase shift characteristics while the 0^{th} derivative did not accentuate phase shift characteristics enough. The 1^{st} derivative SG filter of filter order N=9 was used in the final algorithm, as simulation trials proved it rejected noise in the TCF phase image while maintaining the phase shift characteristics better than the basic difference filter did.

4. One-dimensional Filter Comparisons

a. Derivative Filters

Having completed the second step in the process as shown in Figure 8, the discrete derivative along the TCF phase time axis is computed to further isolate the phase shift while reducing remaining noise distortions. Figure 10 shows clean and noisy (6 dB SNR) TCF phase images where the phase shift time is located at sample 150 in a window length equal to 300 followed by the results of performing the 2^{nd} difference operation and the 1^{st} derivative SG noise filter of length 9 along the time axis. Figures 10.b and 10.c show outputs obtained after applying the differentiation operations to the clean TCF Phase image. Results show the phase shift time information is clearly visible and horizontal lines contained in the rest of the original image are eliminated. However, Figures 10.e and 10.f show that the phase shift time information is lost when these differentiation operations are applied to the TCF phase image obtained from the noisy signal. Simulations showed noise removal is required before applying the differentiation step to preserve the shift timing information.

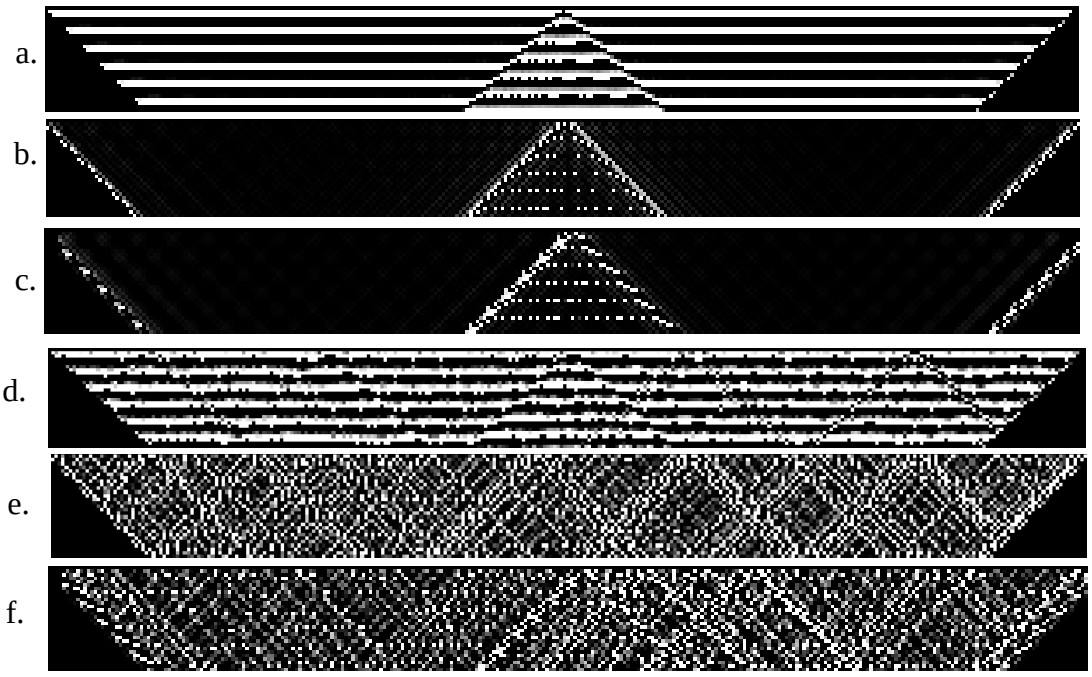


Figure 10. Clean and noisy image difference operations comparisons: a. Clean PSK TCF phase image (shift located at sample 150, window length 300), b. After applying a 2nd Order difference filter along the time axis to image a, c. After applying a 9th Order SG filter along the time axis to image a, d. PSK + AWGN, SNR=6 dB, TCF phase image, e. After applying a 2nd Order difference filter along the time axis to image d, f. After applying a 1st derivative 9th order SG filter along the time axis to image d (Horizontal axis: Time axis – left to right, Vertical axis: Lag axis – top to bottom).

b. Derivative Operation Following One-dimensional Filters Application

Figure 11 shows outputs obtained after first applying 15th order median and mean filters to a noisy TCF phase image (SNR=6dB), and second, applying the 1st derivative SG filter and 2nd difference filter along the time axis. Note that some noise still remains but will be removed by later two-dimensional filtering and morphological operations. Furthermore, the phase shift characteristic located at time 150 in the center of the images is clearly brought out by the median-difference and median-SG combinations as shown in Figure 11.b and 11.c, respectively. Due to the loss of discrete edge values by the mean filter (see Figure 9.) there is no visible phase shift remaining following the derivative operation. The noise filtering of the SG filter brought out only minor amounts

of the phase shift while leaving a considerable amount of noise as shown in Figure 11.f. Overall, the one-dimensional 15th order median filter followed by the 1st order derivative SG filter proved the best option for initial noise removal and phase shift isolation.

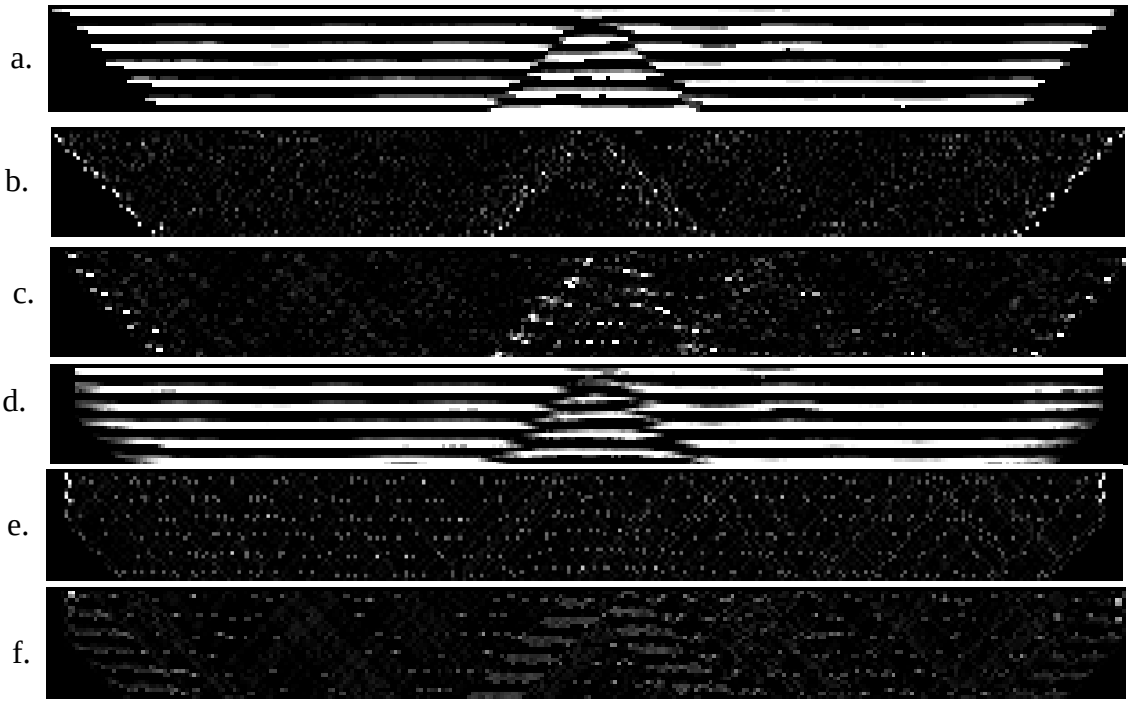


Figure 11. Processed TCF phase images obtained after applying the following filters to TCF Phase output shown in Figure 10.d: a. 1-D 15th order Median along the time axis, b. 2nd order difference along the time axis on image a, c. 1st Order SG filter along the time axis on image a, d. 1-D 15th order Mean along the time axis, e. 2nd order difference along the time axis on image d, f. 1st Order SG filter along the time axis on image d (Horizontal axis: Time axis – left to right, Vertical axis: Lag axis – top to bottom).

Figure 11.c shows that noise distortions still remain in the TCF phase image obtained from PSK signals in 6 dB AWGN. Therefore, two-dimensional filters are applied at that point for additional noise removal, as discussed next, followed by phase shift edge enhancement, and morphological noise removal techniques to produce the final phase shift time image.

B. NOISE REDUCTION: TWO-DIMENSIONAL FILTERS

As discussed in [6], edge detection techniques work reasonably well when images are first pre-processed to mitigate noise effects. Simulations showed that lower order one-dimensional median filters removed noise adequately so that the difference operation enhanced remaining phase shift discontinuities. Simulations also showed that higher order one-dimensional median filters removed too much of the phase shift discontinuity prior to the difference operation. However, the differentiation operation emphasized noise, which was detrimental to the later edge enhancement steps in the TCF phase image processing. Therefore, several two-dimensional pre-processing methods were considered, including Laplace, low-pass, median, and mean filters, to further reduce image noise levels prior to edge enhancement.

1. Lowpass Filter

Lowpass filters, as well as median and mean filters, perform smoothing operations to lessen high frequency details. Note however that we need to preserve long term discontinuities present in the TCF phase information. Equation (4.1) shows two example masks for two-dimensional lowpass filtering. The simplest mask is a matrix of ones, while more complex masks take the form of a three-dimensional Gaussian pulse, where the amount of smoothing experienced is set by the values used in the masks:

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \qquad \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}. \tag{4.1}$$

2. Median Filter

Two-dimensional median filter masks can take on any shape the user wants to define but the generic block mask shown in equation (4.1) above is the most common due to its simplicity. The basic mask is simply a matrix of ones where each item within the region of the mask is sorted and the median value of the group replaces the center or target pixel value. The entire image is filtered when the mask is translated across the image in question performing the median-sort-and-replace operation. Note the resulting

median value tends to be zero when several positive and negative values are collocated. As a result, the median two-dimensional filter may tend to erase all traces of the phase shift in the TCF phase image. This definite drawback was confirmed in our simulations, which showed the two-dimensional median filter was not as useful as the one-dimensional median filter was, and was not used in our final scheme.

3. Mean Filter

The mean filter averages the neighborhood elements within the region covered by the mask resulting in image blurring. Filter orders were kept small, with mask of dimensions 3x3, because the larger the filter order the more blurring and subsequent information was removed along with the noise. Two-dimensional mean filters proved to be extremely good at dampening the noise present in the TCF phase images since sharp details were not as much of a concern as removing the noise. Note that the mean filter maintained phase shift discontinuity information when compared to the amount of noise removed. Equation (4.2) shows a basic mean filter mask:

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}. \tag{4.2}$$

4. Two-dimensional Filter Comparisons

The effects of processing by the two-dimensional filters under investigation is best seen using a TCF phase output image obtained from a PSK signal (6dB SNR) with phase shift at time 150 in the center of a length 300 window previously processed by a 15^{th} order one-dimensional median filter and the 1^{st} order derivative 9^{th} order SG filter, applied along the time axis. Figure 12.a shows the results of the one-dimensional median and SG filters on the noisy image. Figures 12.b and 12.c show that two-dimensional median and mean filters with masks of size 3x3 remove a large amount of noise (represented by the isolated noisy pixels) from the image. However, Figure 12.b shows that the median two-dimensional filter removed a portion of the phase shift time information along with the noise. Figure 12.c shows that the two-dimensional mean filter

removed similar amounts of noise while better maintaining the phase shift time information than the median filter did. Finally, Figure 12.d shows the lowpass filter emphasized unwanted noise that was surrounding the phase shift discontinuity significantly, which is a significant drawback with this filter. Although the phase shift information in the original image of Figure 12.a is decreased by both median and mean filters, the noisy pixels are also removed. Simulations showed this step was necessary for later successful phase shift detection.

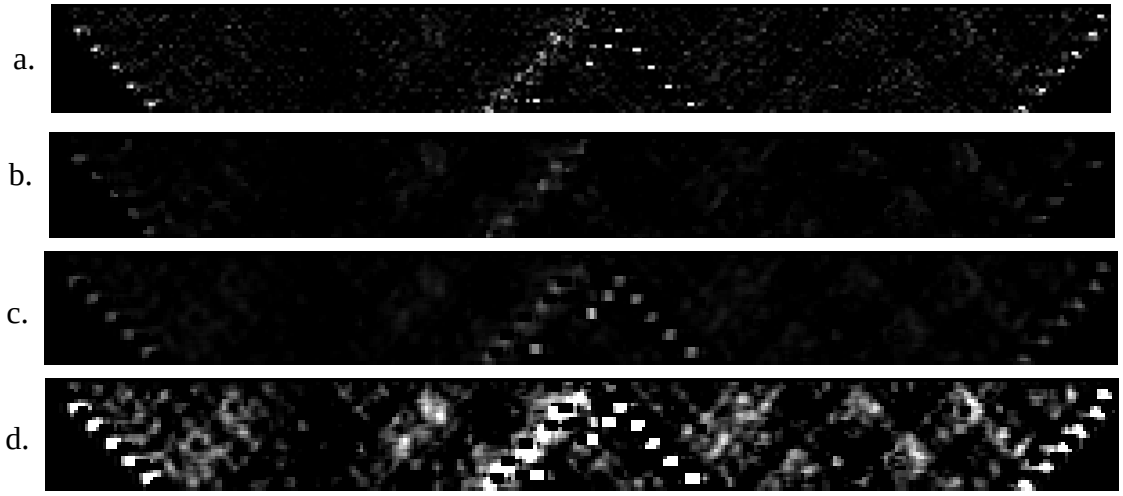


Figure 12. Two-dimensional comparison: a. 1-D Median-SG filtered phase shift image shift time=150, window size=300 before results of two-dimensional 3x3 b. Median, c. Mean, and d. Low pass filters, respectively (Horizontal axis: Time axis – left to right, Vertical axis: Lag axis – top to bottom).

This section discussed specific two-dimensional filters and their definition mask as they apply to noise removal within the TCF phase images. Simulations showed the two-dimensional mean filter discussed is important to remove some image noise prior to step 5 in the image processing flow of Figure 8. Simulation showed the two-dimensional mean filter of size 3x3 discussed was better at removing noise distortions while preserving phase shift time information than the other filters were. Next, we discuss the edge detection operation (step 5 of the processing flow shown in Figure 8).

C. EDGE DETECTION

Edge detection schemes are investigated to isolate the cross-term region boundaries discussed in Chapter III. We specifically consider Laplacian, Roberts, Sobel and Prewitt edge operators and variants.

1. Laplacian Operators

Two example masks for the Laplacian operator are shown below in Equation (4.3) [[6] p. 52,66]. The first effectively sharpens image details while the second is an edge detection mask. In general, these filters sharpen the image well but accentuate image noise as well as discussed later. Note that 3x3 masks are shown below, however, larger masks tend to mitigate the effect noise has on the operation:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} \qquad \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}. \tag{4.3}$$

2. Roberts Edge Operator

The Roberts edge operator is a differentiator style operator that compares neighboring pixels through a difference of squares comparison. One expression given in [7] is as follows:

$$g(x,y) = \left\{ \left[\sqrt{f(x,y)} - \sqrt{f(x+1,y+1)} \right]^2 + \left[\sqrt{f(x+1,y)} - \sqrt{f(x,y+1)} \right]^2 \right\}^{1/2}, \tag{4.4}$$

where *f(x,y)* represents the image undergoing the operation with integer pixel coordinates *(x,y)*. Thus, regions where neighboring pixels greatly differ are emphasized more than regions with small pixel intensity differences. While Equation (4.4) is one version of the Roberts edge detection operator, as discussed in [6], other versions exist. For example, [6] proposed a simpler expression of the Roberts edge detection operator defined in terms of absolute values instead of square roots for the same neighboring pixels as in Equation (4.4) as follows:

$$g(x,y) = \left| f(x,y) - f(x+1,y+1) \right| + \left| f(x+1,y) - f(x,y+1) \right|. \tag{4.5}$$

3. Sobel Edge Operator

As discussed above, the Roberts edge operator filters the image by comparing neighboring pixels to each other to enhance edges present in the image. The Sobel edge operator uses masks instead to perform a filtering operation based on predefined directions. Thus a Sobel mask is designed to find edges oriented vertically (column mask), horizontally (row mask), or diagonally (diagonal mask). Therefore, the user is able to apply specific masks to highlight certain edges vice others. Note that such a tool where edges with known orientations point to the phase shift times of interest may be potentially very useful.

Examples of column, row, and diagonal Sobel operator masks used in this research are shown in Equation (4.6). These masks were extended to higher orders, i.e. 7x7, to investigate how mask dimensions affect results and will be discussed below in Section 5.

$$\underset{\text{COLUMN MASK}}{\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}} \quad \underset{\text{ROW MASK}}{\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}} \quad \underset{\text{RIGHT DIAGONAL MASK}}{\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 2 & 0 & -1 \\ 1 & 0 & -2 & 0 \\ 0 & -1 & 0 & 0 \end{bmatrix}}. \tag{4.6}$$

It should be mentioned that the right diagonal mask shown in Equation (4.6) is a version of the Robinson compass mask listed here because of the similarity to Sobel operators but emphasizing a directionality that the normal Sobel operator does not use [6]. Results show the Robinson compass mask shown yields better results for pixels oriented at a positive forty-five degree angle, while column and row masks provide better results for pixels oriented vertically and horizontally.

4. Prewitt Edge Operator

Prewitt masks are similar to Sobel masks in shape and orientation but use only ones and zeros (Equation 4.7). Sobel masks allow for variations in the coefficient magnitudes (Equation 4.6) whereas Prewitt masks do not. The Prewitt operator is not as

drastic an operation as the Sobel operator because the Prewitt operator does not emphasize any one pixel within its masks unlike the Sobel operator. Subsequently, the Prewitt operator performed worse than the Sobel in the noisy environment of our application.

$$\underset{\text{COLUMN MASK}}{\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}} \quad \underset{\text{ROW MASK}}{\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}}. \tag{4.7}$$

5. Edge Operator Comparisons

a. Noisy Phase Shift Example

Comparing edge operators was somewhat more subjective than other comparisons conducted in this study; however, examples show which operators work best for this project's requirements. Figure 13 illustrates the impact edge detection operators have on the TCF phase obtained for a PSK signal imbedded in 6dB SNR level noise after the one-dimensional 15^{th} order median and 1^{st} derivative 9^{th} order SG filters discussed previously have been applied. The phase shift is located at time 150 in the window of length 300. Figure 13 also shows that three of the four edge operator schemes have very similar results with the Roberts edge detection slightly better than the others. Sobel and Prewitt operators tended to emphasize noise pixels more than the Roberts edge operator did, as shown in Figure 13.c and 13.d. The two-dimensional Laplacian filter did not perform well, as shown in Figure 13.b, and was not considered further. The Roberts edge detection operator was selected in our study, as simulations showed it emphasized fewer noise pixels than the other edge detection schemes.

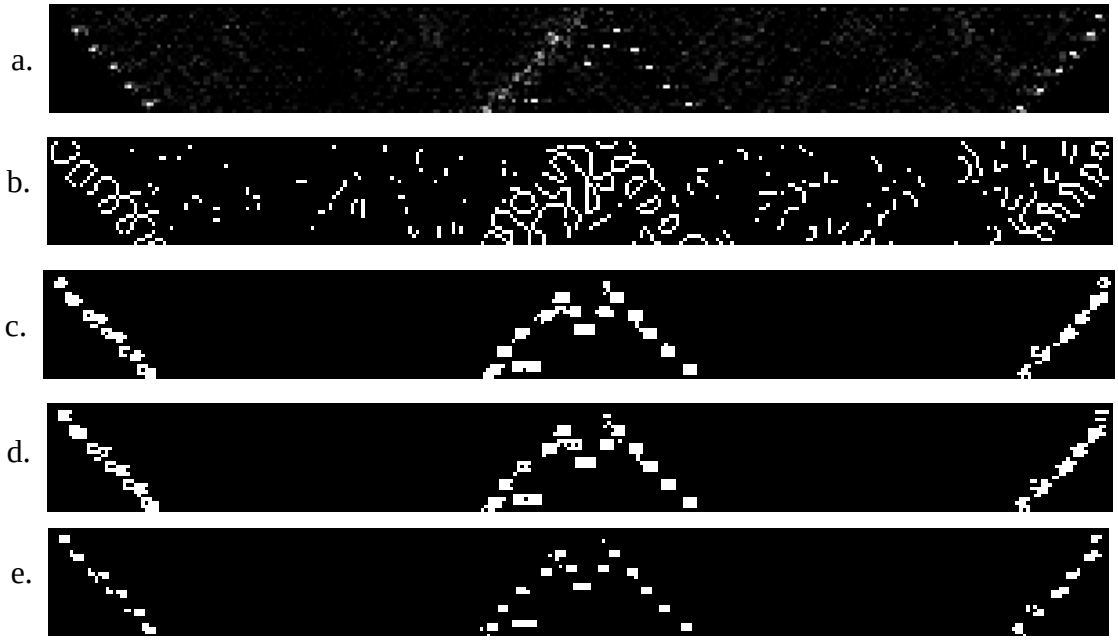


Figure 13. Edge Operator Comparisons: a. Processed TCF phase image obtained after SG filter operation, filtered image obtained after b. Laplace, c. Sobel, d. Prewitt, and e. Roberts edge detection schemes (Horizontal axis: Time axis – left to right, Vertical axis: Lag axis – top to bottom).

b. Noisy No Phase Shift Example

All edge detection schemes are sensitive to image noise. Therefore, the no-phase-shift case must also be taken into consideration when deciding which edge detection method to use for image enhancement purposes. Figure 14.a plots the TCF phase obtained for a PSK signal without a phase shift imbedded in 6dB SNR. Figures 14.b through 14.e showed that the Sobel, Prewitt, and Roberts edge detection schemes perform similarly in this no-phase shift example. Overall simulations showed that the Roberts edge detection scheme yields smaller noise pixel regions than those obtained after applying the Sobel or Prewitt operators and minimize the noise remnants to be cleaned later. As a result, the Roberts operator was selected to emphasize the phase shift characteristics prior to the final morphological image transformation phase.

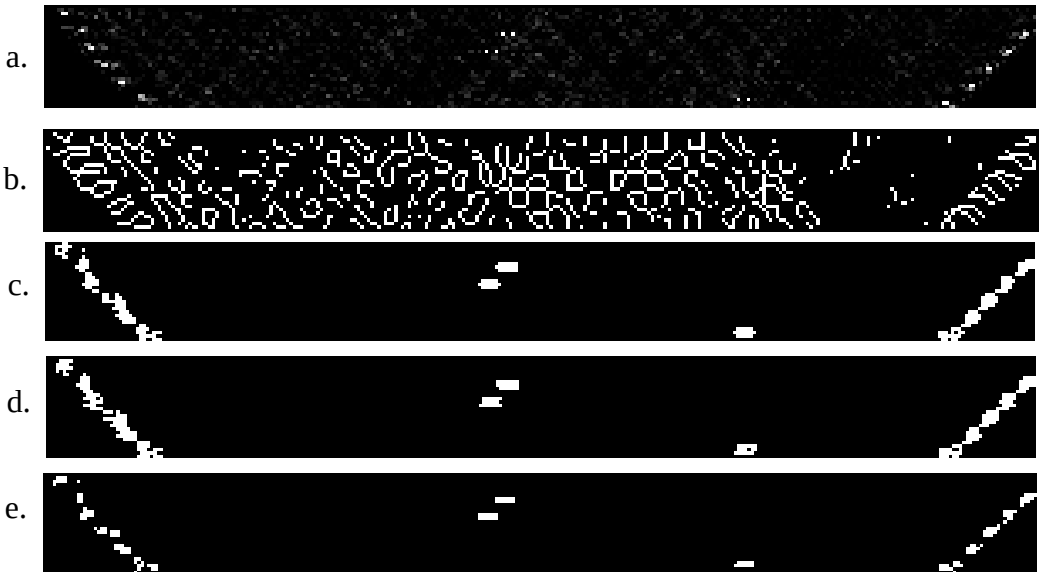


Figure 14. No Phase Shift Edge Comparison: a. Processed TCF phase image obtained after the SG filter operation, results of b. Laplace, c. Sobel, d. Prewitt, and e. Roberts edge detection schemes (Horizontal axis: Time axis – left to right, Vertical axis: Lag axis – top to bottom).

This section discussed four different edge detection schemes designed to enhance region boundaries containing the phase shift timing information (step 5 of Figure 8). At this point the TCF phase output image is filtered along the time axis with a 15^{th} order one-dimensional median filter, a 2^{nd} order difference or 1^{st} derivative 9^{th} order SG filter, and further filtered using a two-dimensional 3x3 mean filter. Next, the Roberts edge detection scheme is applied to emphasize phase shift time information. Finally, morphological operations are applied as a sixth and final pre-processing step, and resulting processed image used to extract the phase shift timing information. Morphological operations considered in this study are discussed next.

D. MORPHOLOGICAL OPERATIONS

The phase shift characteristic is more clearly defined in a phase shift window (Figure 13.e) after the edge enhancement step is completed than it originally was in the initial TCF phase output image (Figure 9.a). However, a pixel pattern consistent enough to be used for a matched filter approach does not exist at this stage. Additionally, some noise remains in the images following the edge detection step. This section introduces

morphological dilation and erosion techniques and discusses their uses to consistently transform the edge enhanced TCF phase image into an image that is exploitable by a matched filter approach while eliminating noise remnants.

1. Dilation and Erosion Basics

Dilation, erosion and subsequent binary opening and closing operations were considered in this study. Erosion and dilation, as discussed in [8] and [6], serve as the fundamental functions defining opening and closing operations. Erosion is defined in [6] as a logical NOR operation where image pixels covered by the structuring element are replaced by taking the logical NOR of all pixels under consideration. The logical NOR of the pattern under the structuring element shrinks the eroded object or pattern when the structuring element does not fit inside the object being considered.

Dilation is defined as a logical OR operation where the region of the object within the structuring element object is replaced by the logical OR of the structuring element samples unless the origin pixel is a zero [6]. Dilation is the dual of erosion, as explained in [8], and has the effect of expanding objects under consideration. Furthermore, a structuring element is a predefined pattern of pixels that must be chosen for each specific case in order to provide the desired result. Thus, these operations are generally very subjective and do not lend themselves easily to our specific case. However, they were considered as means to isolate key features of the TCF phase output images to improve the phase shift time detection task. Due to the nature of the processing used these operations were considered primarily as a way to remove noise pixels still remaining after step 5 of the pre-processing phase shown in Figure 8.

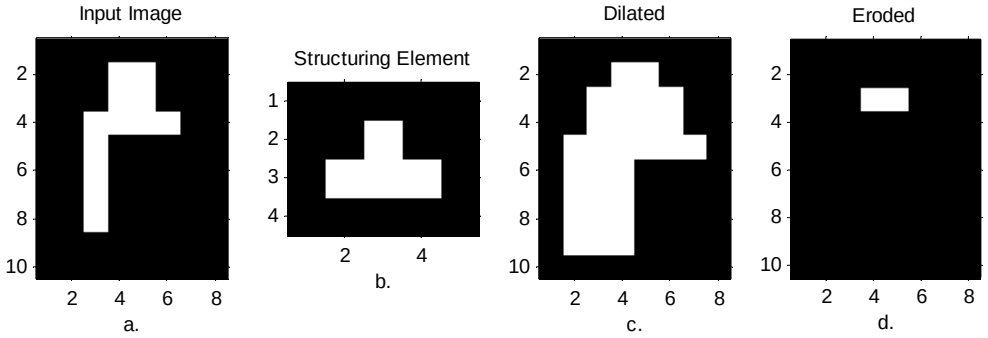


Figure 15. Binary Dilation and Erosion Example: a. Input image, b. Structuring element, c. Dilated image, d. Eroded image.

A simple binary image example using a sample object and sample structuring element illustrates dilation and erosion effects for our purposes. Figure 15 shows the results of separate dilation and erosion operations on the white object contained in Figure 15.a using the upside down 'T' structuring element shown. Note that the structuring element in Figure 15.b is said to have dimensions of 4x5 pixels, while the structuring element object within the image is 3 pixels along the base and 1 pixel above the base at the center. This distinction between images and the objects they contain is important for the current discussion and later chapters. Note how the dilation effectively expands the irregular object, while the erosion operation almost entirely erases the object. Further note that the erosion operation may completely erase the object when the structuring element used is too large or does not fit within the object of consideration due to geometry or orientation.

Dilation and erosion methods will be used to our advantage for noise removal as well as shaping the TCF phase shift characteristic. However, the user must be careful when choosing structuring elements especially with the erosion operator as phase shift features must be preserved. Too large erosion structuring elements result in missed detects, and too small a structuring element results in false alarms. Considering the peculiarities of dilation and erosion techniques and after trial and error, we selected specific irregularly shaped structuring elements of different sizes, and cascaded dilation and erosion operations to accomplish our dual goals of phase shift characteristic

transformation and noise removal. Although project templates are discussed in detail in Chapter V, basic structuring elements are used here to illustrate the dilation and erosion process.

2. Morphological Comparisons

Morphological operations, shown as step 6 in the image processing flow, transforms the image shown in Figure 13.e into the final image shown in Figure 7. Since the sides of the characteristic phase shift time triangle shown in the final image of Figure 7 were the desired result of these morphological operations, the following discussions illustrate how changing structuring elements may affect the resulting phase shift characteristic shown in Figure 13.e. The final decision to use cascaded dilation-erosion operations came after considering the initial images of Figures 13.e and 14.e and the final desired image of Figure 7 as well as the simulation results discussed below.

a. Square Structuring Element Dilation and Erosion

We applied dilation and erosion operators to TCF phase images obtained from PSK signals imbedded in AWGN with SNR=6dB and already pre-processed thru step 5 of Figure 8, i.e., including the Roberts edge operator to investigate the impact of the operations. Note that PSK signals with phase shift or no phase shift contained in the frame analyzed were considered. Figure 16 illustrate these operations on a pre-processed TCF phase image for a phase shift case. The morphological operation demonstrated here shows that dilation or erosion operations independent of one another do not enhance the phase shift information or enhance both phase shift information and noise. Note the structuring element objects for these dilation and erosion operations are 4x4 filled-in squares. Figure 16.b shows that dilation produced amplification of the desired triangular phase shift at time 150 but also amplified noise present elsewhere in the image. Furthermore, Figure 16.c shows that the erosion operation on Figure 16.a erases the entire image due to the characteristic size and shape of noise pixel groups present in this figure. Thus, dilation and erosion operations conducted independently are not successful at emphasizing phase shift information and noise removal.

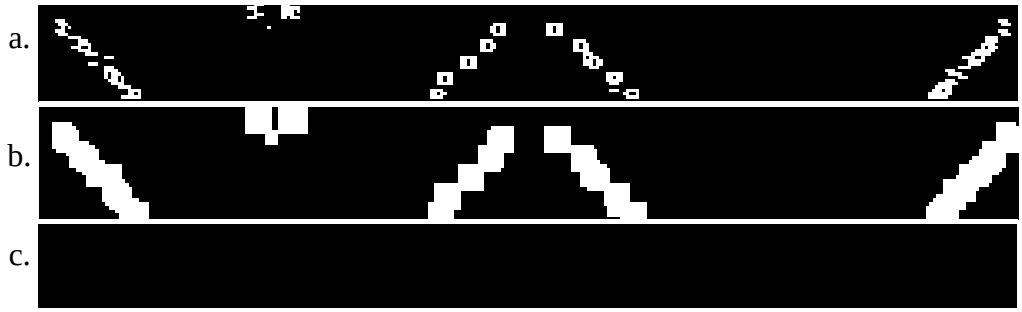


Figure 16. Impact of independent dilation and erosion of TCF phase image with, phase shift example for PSK signal +AWGN, SNR=6dB (shift time 150, window length=300): a. after applying Roberts edge detection, b. after applying dilation operation with 4x4 square structuring element on image a, c. after applying erosion operation with 4x4 square structuring element on image a (Horizontal axis: Time axis – left to right, Vertical axis: Lag axis – top to bottom).

b. Closing and Cascade Examples

By definition a closing or opening operation uses the same structuring element for both dilation and erosion portions. However, simulations showed that dilation and erosion operations did not remove unwanted noise pixels when using identical structuring elements for both operators. Figure 17 illustrates this problem, where the circled region indicates unwanted noise pixels remaining after the two operations. Therefore, specific structuring elements were designed to transform the sequences of pixels showing at $\pm 45°$ angles into continuous lines along the same orientations, as shown in Figure 7.

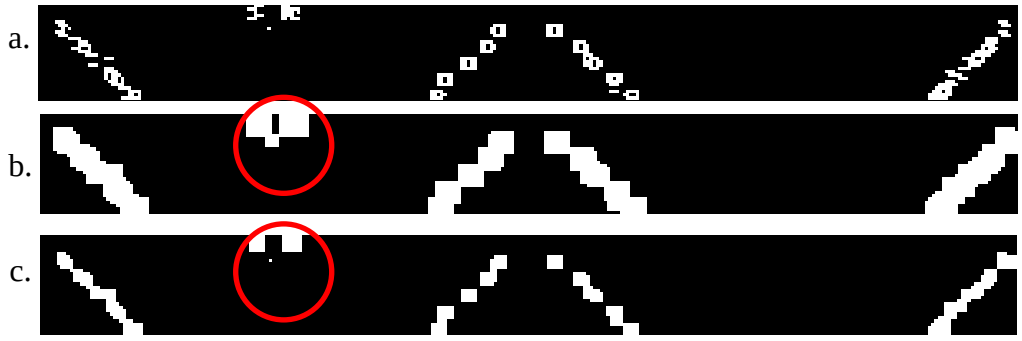


Figure 17. Closing operation example (shift time 150, window length=300): a. pre-processed TCF phase image (with phase shift, PSK signal + AWGN, SNR=6dB), Roberts edge detection output, b. after applying Dilation operator, 4x4 square structuring element, c. after applying erosion operator to image b, 4x4 square structuring element (Horizontal axis: Time axis – left to right, Vertical axis: Lag axis – top to bottom).

c. Irregular Structuring Elements

Two sets of structuring elements were used to clean up noise and preserve lines oriented at $\pm 45°$ angles that contain the phase shift timing information. Figures 18 and 19 show the first and second sets of structuring elements used to accomplish the desired final image transformations. Note that the dilation template called 'Dilation 1' is 12x6 pixels, while each erosion template in Figure 18 is 24x24 pixels. The final dilation and erosion templates are 8x8 and 11x6 pixels respectively, as shown in Figure 19. The size difference between the erosion and dilation structuring elements within each set is required to facilitate the noise removal feature of the cascade operation during the both operations. Furthermore, the first set is used to aid the phase shift triangular shape construction from the edge detection phase output. These erosion operations are conducted separately then summed together to result in a relatively noise free image with the phase shift characteristic more pronounced than that observed before their applications, as shown in Figure 20. The second set of structuring elements was used to complete the transformation by fattening the phase shift lines to maximize correlation, while at the same time sharpening the phase shift characteristic object and removing noise emphasized by the second dilation operation.

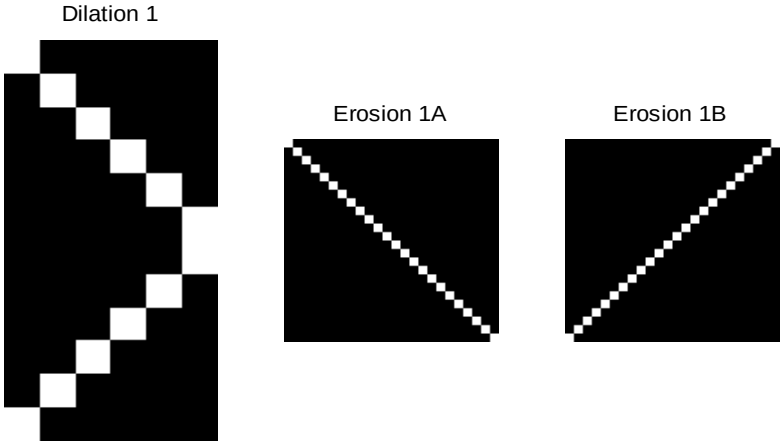


Figure 18. First irregular structuring element set for Dilation and Erosion (Sizes: "Dilation 1": 12x6, "Erosion 1A" & "1B": 24x24).

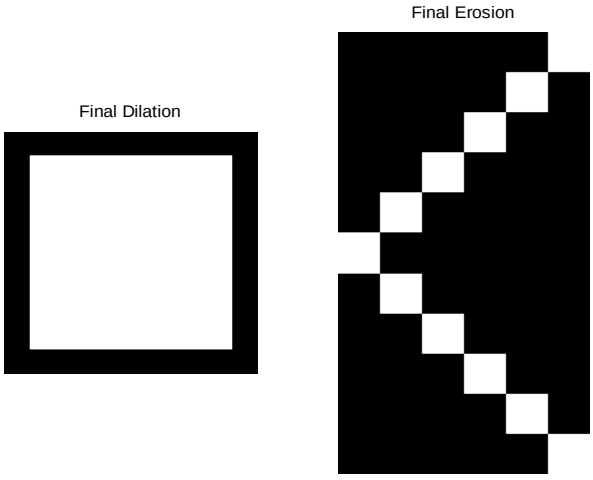


Figure 19. Final set of irregular structuring elements (Sizes: “Final Dilation”: 8x8, “Final Erosion”: 11x6).

Figure 20 presents results obtained after applying the two structuring element sets shown in Figures 18 and 19. Figure 20.e shows the result of the first set where the desired phase shift characteristic remains. Figure 20.g shows the final processed image following the application of the second structuring element set, where the phase shift characteristic is enlarged. Thus, two successive cascaded dilation-erosion operations using irregularly shaped structuring elements performed the desired transformation and simultaneous noise removal yielding a final image similar to that shown in Figure 7.

Figure 20. Cascade morphological example using irregular structuring elements (shift time 150, window length=300): a. PSK + AWGN, SNR=6dB, TCF phase-shift edge detection output, b. Input dilated using 'Dilation 1' structuring element, c. Dilated image eroded using structuring element 'Erosion 1A', d. Dilated image eroded using structuring element 'Erosion 1B', e. Eroded images summed to form composite processed image f. Set 1 eroded image dilated using 'Final Dilation' structuring element, g. Final dilated image eroded using 'Final Erosion' structuring element (Horizontal axis: Time axis – left to right, Vertical axis: Lag axis – top to bottom).

This section investigated dilation, erosion, and binary closing and their application to the pre-processed TCF phase output. We designed specific structuring elements, shown in Figures 18 and 19, to enhance the phase shift timing information and showed in an example the impact these specific structuring elements have on noise pixels and phase shift timing information contained in the TCF phase image.

E. CONCLUSIONS

Several different techniques were considered to clean up the TCF phase matrix and enhance the phase shift timing information contained within. First, we applied a one-dimensional median filter along the time axis to remove image noise without destroying the phase shift discontinuity information. Next, we applied a differentiation step to enhance the phase shift discontinuity. Third, we applied a two-dimensional mean filter of dimension 3×3 to remove remaining noise from the TCF phase image. Fourth, we used the Roberts edge operator to enhance the phase shift time information. Last, morphological feature manipulation techniques including binary dilation and erosion were applied to further improve the detection results of the overall algorithm.

Overall, findings showed that,

- the one-dimensional median filter removed enough noise from the TCF phase image to emphasize any present phase shift discontinuities after the differentiation step,
- the two-dimensional mean filter proved very useful at removing image noise following the differentiation step,
- the Roberts edge detection operator was best at enhancing phase shift timing information among the four edge operators investigated,
- cascaded dilation and erosion operations using irregularly shaped structuring elements, as those shown in Figure 18 and 19, can effectively remove remaining image noise features while preserving phase shift features.

The basic sequence of operations applied to the TCF phase image to enhance phase shift time information is summarized below using a specific PSK noisy signal example. The PSK signal considered has a phase shift at time=150 and SNR level equal to 6dB. Figure 21 shows the resulting TCF phase image obtained from that signal.

Figure 21. QPSK + AWGN, SNR=6dB, TCF phase output and phase shift time at 150 (Horizontal axis: Time axis – left to right, Vertical axis: Lag axis – top to bottom).

(1.) Apply a one-dimensional median filter of length 15 along the time axis to mitigate noise affects,

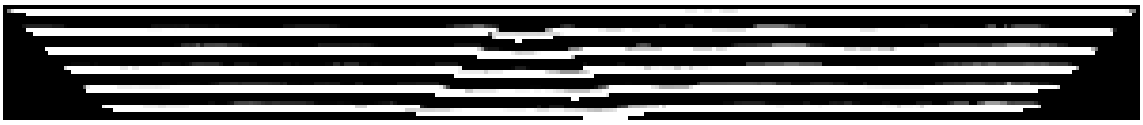


Figure 22. One-dimensional filtered TCF phase output of Figure 21 (Horizontal axis: Time axis – left to right, Vertical axis: Lag axis – top to bottom).

(2.) Apply a 2^{nd} order difference filter or 1^{st} derivative 9^{th} order SG filter along the time axis to emphasize any phase shift discontinuities across the time axis,

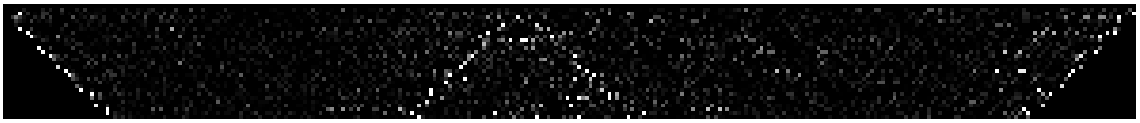


Figure 23. TCF phase Output of Figure 22 following 2^{nd} order difference operation along time axis (Horizontal axis: Time axis – left to right, Vertical axis: Lag axis – top to bottom).

(3.) Apply a two-dimensional mean filter of dimension 3×3 to further remove remaining noise while maintaining the phase shift information,

Figure 24. Figure 23 following two-dimensional mean filter of order 3 (Horizontal axis: Time axis – left to right, Vertical axis: Lag axis – top to bottom).

(4.) Apply the Roberts edge operator to emphasize any phase shift information. Note this step also converts the image from grayscale to binary,

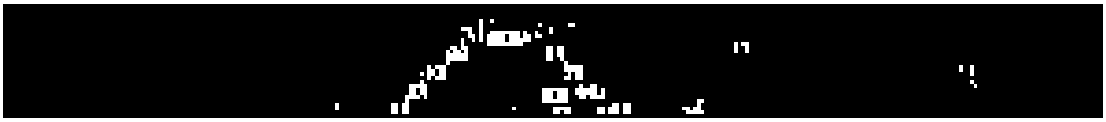


Figure 25. Figure 24 following Roberts edge detection operation (Horizontal axis: Time axis – left to right, Vertical axis: Lag axis – top to bottom).

(5.) Apply cascaded binary dilation and erosion operations previously discussed in Section D.2.c to remove remaining random noise and emphasize phase shift timing information.

Figure 26. Final image following cascaded Dilation and Erosion operations on Figure 25 (Horizontal axis: Time axis – left to right, Vertical axis: Lag axis – top to bottom).

THIS PAGE INTENTIONALLY LEFT BLANK

V. PHASE SHIFT DETECTION ALGORITHM

The overall phase shift time detection algorithm can be split into two phases: (1) a two-dimensional cross correlation matched filter operation and (2) the optional window overlap phase shift decision scheme designed to improve performances in noisy conditions. This section details the algorithm implementation.

A. PHASE SHIFT DETECTION ALGORITHM IMPLEMENTATION

Figure 27 presents the overall flow chart of the major steps taken within the phase shift detection algorithm designed during this work. As shown below, first target images and structuring elements are defined (Chapter IV) followed by the Temporary Correlation Function (TCF) phase computation for each signal segment (Chapter III). These TCF phase images are then processed to isolate the phase shift time (Chapter IV) before the two-dimensional cross-correlation locates the phase shift time in each window. Last, final phase shift time decisions are derived, either from individual decisions, or by combining multiple decisions when overlapping analysis windows is allowed during processing. Topics that have been previously discussed will refer to earlier sections of the text with specific details of the implementation briefly discussed in the following section. The remainder of this section describes in detail each phase not previously covered.

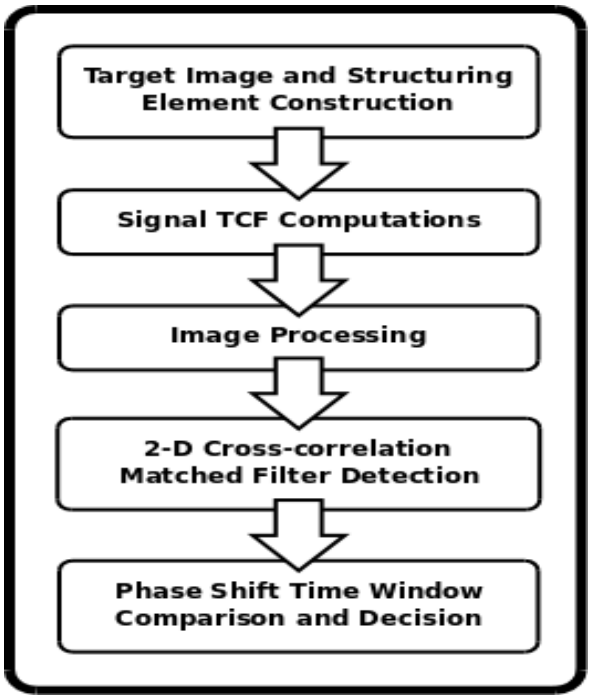


Figure 27. Phase Shift Detection Algorithm flow diagram.

1. Signal Generation

As stated in Chapter II, this research focuses on identifying phase shift times contained in noisy Quadrature Phase Shift Keyed (QPSK) signals. The signal generation was conducted so that it minimizes assumptions about the incoming signal. Some assumptions were inherent such as using normalized frequency less than 0.5 Hz to ensure the Nyquist criterion is not violated. The phase shifts used were of the same magnitudes as those discussed in Chapter II, Equation (2.2).

The last significant assumption that is not addressed for this research is the affect of bit periods on this algorithm. The phase shift insertions are programmed in this implementation to allow one, zero, or at most two shifts to be present within each analysis frame. Additionally, the spacing of the phase shifts within the signal is dependent on the number of desired test shifts and signal length simulated. Signal lengths and number of shifts within a signal simulation run were calculated to prevent

more than one phase shift from occupying the same window during no overlap runs and more than two shifts during overlap runs. Thus, a minimum bit period was not analyzed as part of this work.

2. TCF Phase Image Computation

a. Hilbert Transform

The TCF phase computation performs a complex transformation on the input signal but requires an analytic form of the real signal shown in Equation (2.1). The signal generation for this research is a real cosine signal waveform, as presented in Chapter II. The real signal is transformed into its analytical version using the Hilbert transform defined in the frequency domain as:

$$\begin{aligned} H_{HB}(\omega) &= \begin{cases} -j, & \text{for } \omega > 0 \\ 0, & \text{for } \omega = 0 \\ j, & \text{for } \omega < 0 \end{cases} \\ &= -j\,\mathrm{sgn}(\omega). \end{aligned} \tag{5.1}$$

In this study, we apply the MATLAB function *hilbert.m* to the real signal to generate the analytical signal.

b. TCF Phase Calculation

The resulting analytic signal obtained after the Hilbert transform step is used to compute the two-dimensional TCF phase function previously discussed in Chapter III.

c. Phase Angle Computation

Raw TCF expressions must have the phase information extracted using a phase angle computation. This work used the MATLAB function *angle.m* to compute the phase angles following the TCF computations.

3. Target Image Generation

Many trials were conducted to find appropriate target templates needed for the two-dimensional matched filter operation. Simulations showed that too small template images tended to highly correlate with any bit of remaining image noise, while too large template images did not significantly correlate with present phase shift characteristics. These correlation issues will be discussed in detail in Section C below. Final target images, shown in Figure 28, were produced in MATLAB using matrices of ones and zeros and then stored for recall by the program for each trial run.

Figure 28. Target images used during 2-D Cross-correlations (Horizontal Axis – Time Axis, Vertical Axis – Lag Axis).

Target images have dimensions 60x30 pixels along the time axis and lag axis, respectively, white target objects thickness equal to 7 pixels. These two target images were selected as pre-processed TCF phase images obtained after filtering and morphological operations reflected similar structures: triangular shape with closed (as for target 1) or open tip (as for target 2) portions.

Further, note that the thickness of the triangle object present in both target images is indirectly a result of the size and shape of the structuring elements used in the morphological operations. We noted that this specific thickness parameter could adversely affect the two-dimensional matched filter results if not chosen properly. We selected the specific 7-pixel object thickness empirically after varying this parameter and observing resulting phase shift and noisy no-phase shift correlation values.

4. Signal Image Pre-processing Details

Throughout this section, the sample TCF phase shift image shown in Figure 29 obtained from a 6 dB noisy QPSK signal is used to illustrate each processing step. The length of data in this window segment is 364 samples to illustrate the final windowing

method used in the algorithm. The TCF phase image dimensions are 364 pixels long by 182 pixels wide before processing begins. The phase shift time in this example is located at time 150, as indicated in Figure 29. Note the phase shift is not visible form the initial TCF phase image.

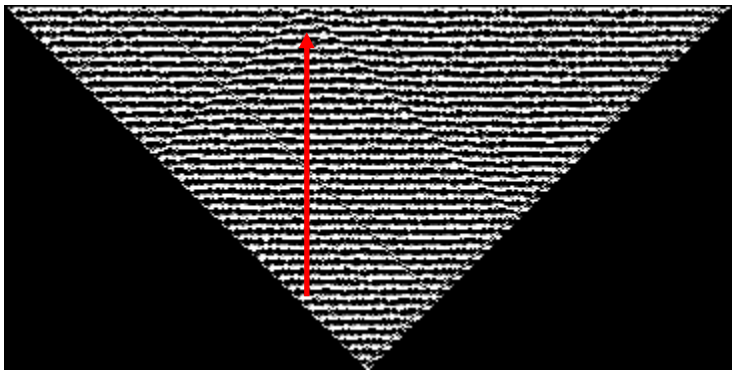


Figure 29. PSK + AWGN, SNR=6dB, TCF phase shift image prior to image processing (arrow indicates shift time=150, for window length=364) (Horizontal axis: Time axis – left to right, Vertical axis: Lag axis – top to bottom).

a. Signal Segmentation

Detection results are directly impacted by the signal segmentation process, and the TCF generation (Chapter III). The following comments can be made.

- Simulations showed that phase shifts located towards signal extremities were more difficult to extract, as a result of the pre-processing steps implemented to decrease noise and emphasize phase shift timing information.
- The TCF phase function displays information for signal times less than or equal to the lags calculated, thus resulting in the triangular shape observed in Figure 29, i.e., leaving a portion of each end of the time axis blank, and generating outside edges in the TCF phase image. These TCF outside boundaries do not contain phase shift timing information, and are removed by masking the first and last 32 pixels on either end of the TCF phase output image to eliminate them. However, such masking also results in

the detection algorithm un-ability to detect potential phase shift information contained within these time segments, as illustrated in Figure 30.

Two different segments options were considered in this work:

- Segmentation Option A: Segment the PSK signal using the desired window length or frame size, as shown in Figure 30. This approach is simple but leaves a segment of the data un-analyzed, thereby missing any phase shift contained within these end regions.
- Segmentation Option B: Expand the PSK window length by 32 samples on either side of the frame to compensate for the masking operation of the same width, also illustrated in Figure 30. The additional 32 samples added on either side ensure all portions of the signal are visible in the cross-correlation computation and no segment of the signal is left un-analyzed.

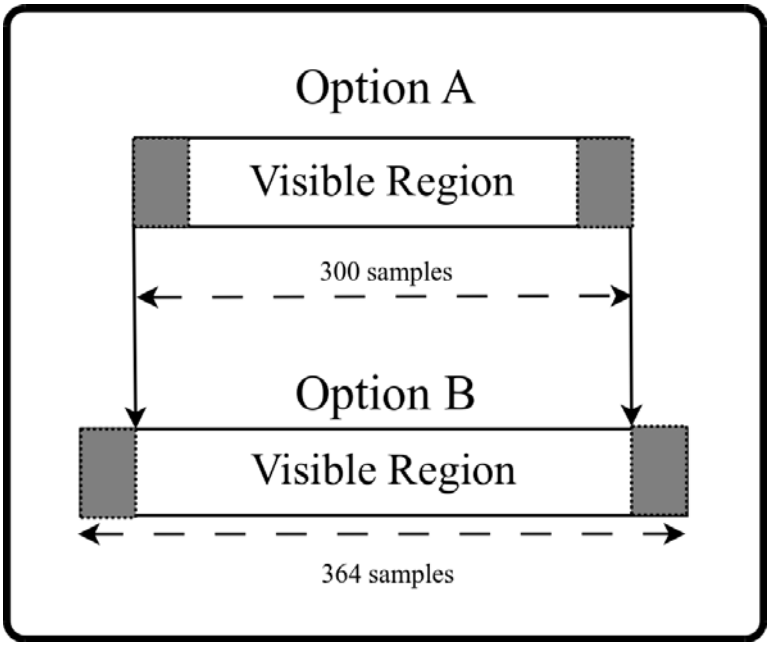


Figure 30. Segmentation Options A and B.

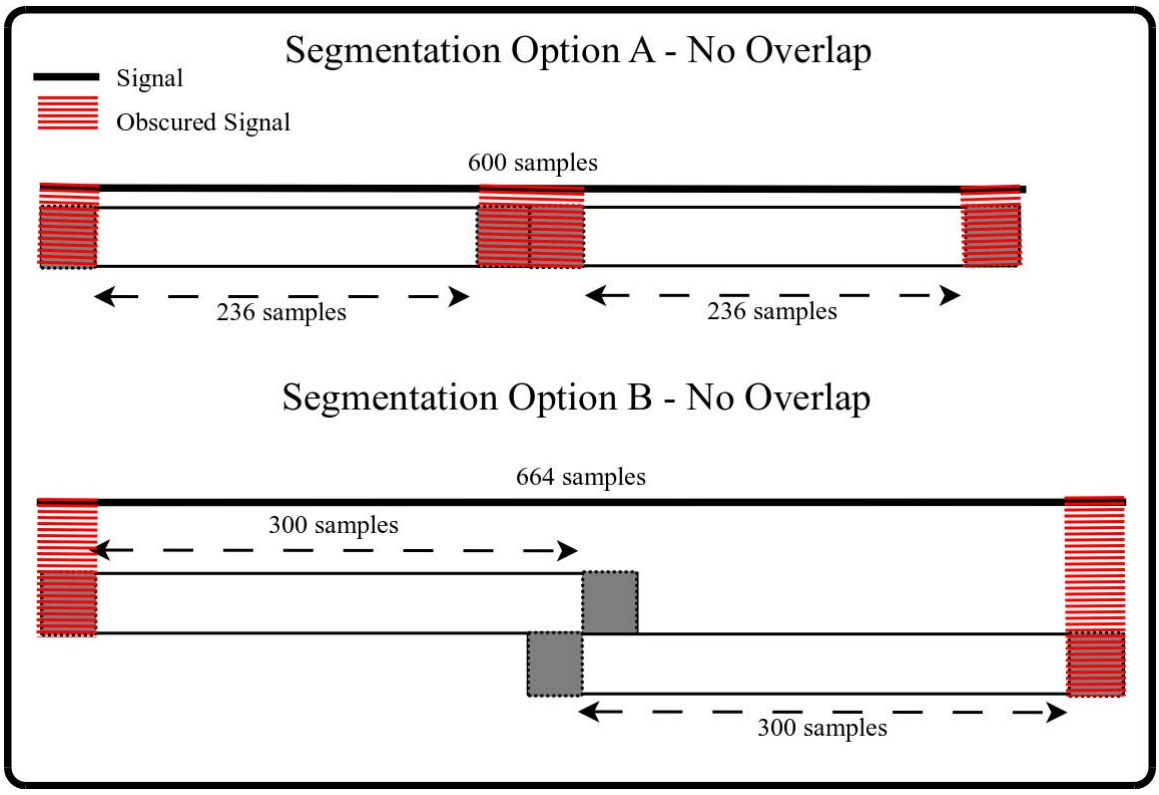


Figure 31. Signal Segmentation and Masking.

b. Image Construction

The TCF transformation is applied to each signal frame to obtain the TCF phase output image. The resultant size for a window length of 300 samples using segmentation option B is 364 pixels long by 182 pixels wide, as the TCF phase image length is dictated by the overall signal frame length. The width of the signal image is dictated by the TCF function itself and is equal to half the total signal length used in the TCF transformation operation. The TCF computation performs the temporal correlation for positive lag values only, as the function is complex symmetric.

c. Image Cropping

Simulations showed the processed TCF phase images could not be processed successfully using the default TCF image size because the target images matched the signal images at multiple locations, as illustrated in Figure 32. Note that

Figure 32 shows the processed TCF phase output obtained after filtering and morphological operations were applied to the raw TCF phase shown in Figure 29. Further, note that the phase shift time information is clearly visible at that point. This problem stemmed primarily from the Target image correlating with features of the signal image that closely but incorrectly matched the actual phase shift characteristic, as described in point number 3 of Figure 32. Therefore, the processed TCF phase images were resized to the same width as that of the target images, i.e., signal images were cropped to a width of 30 pixels as selected target images had dimension 60 by 30 pixels to avoid such errors.

The primary focus of the phase shift time estimation algorithm is to find the shift time characteristic "tip" of the triangle located along the time axis. This desired triangular feature is what the target images discussed in Section 3.a were based upon. Smaller target images better focused on this key phase shift characteristic negating the need for larger images. Significant computational savings were realized through reduction of the signal image size, thereby limiting the two-dimensional cross-correlation computation to one pass down the image vice 182 passes for the un-cropped version, as shown in Figure 32. Also, larger target images provided fewer correlation values during the cross-correlation process because only those values that resulted from the correlation of the full target image inside the signal image were accepted. Thus, larger target images were shown through simulation to increase the number of missed phase shifts for a given window length and unnecessarily increase the computational load.

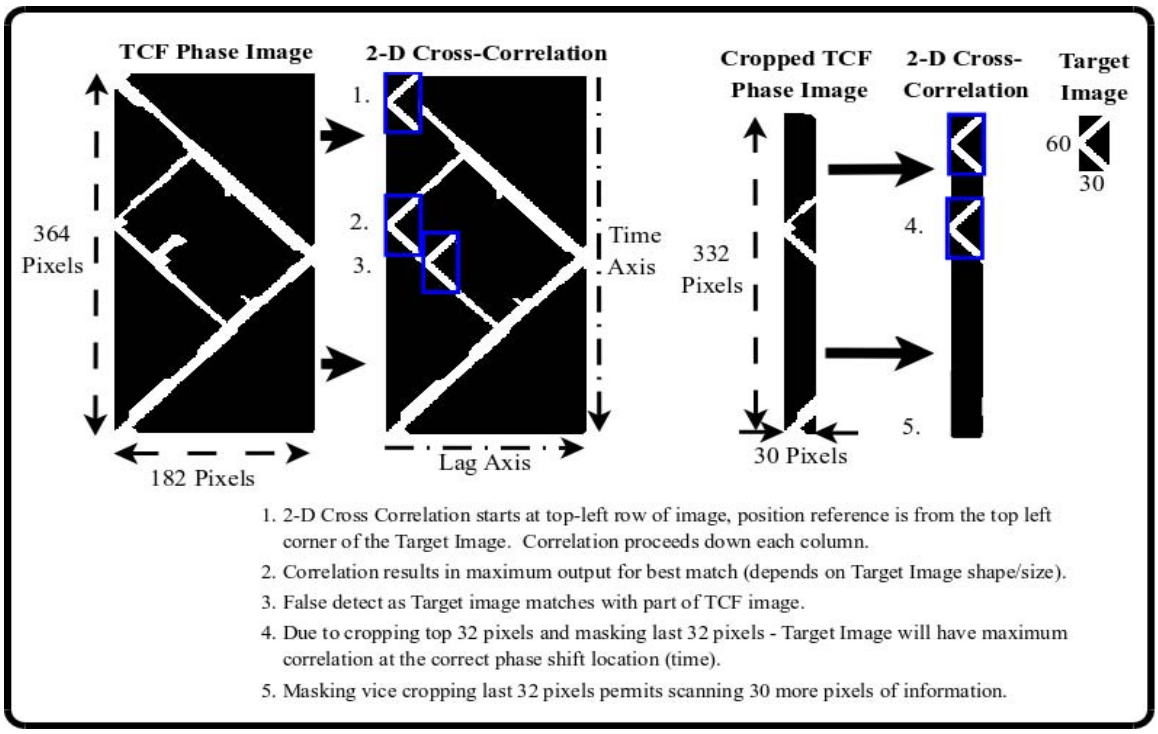


Figure 32. 2-D Cross-Correlation Example: 1.) Processed TCF phase image before and after cropping, 2.) Showing correlation process using Target image (All images: Vertical axis: Time axis – top to bottom, Horizontal axis: Lag axis – left to right).

Figure 32 shows the original un-cropped and cropped versions of the TCF phase image. Note that image sizes indicated within the figure illustrate changes in image size resulting from the image cropping step. Specifically, the image cropping step converts the processed TCF image from 364x182 pixels to 332x30 pixels by removing a rectangular block containing the first 32 points along the time domain in the processed TCF image. The trailing 32 pixels time-domain pixels are masked to allow the two-dimensional cross-correlation step a longer scan of the signal image, as explained in Section 4.j. Figure 33 shows the results of the cropping operation illustrated in Figure 29.

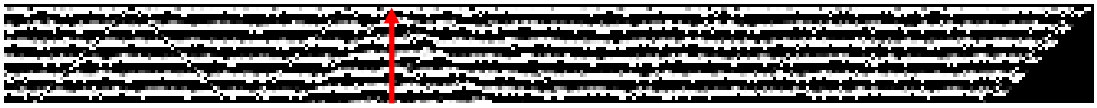


Figure 33. Example cropped TCF phase output image for QPSK signal + AWGN, SNR=6dB (Horizontal axis: Time axis – left to right, Vertical axis: Lag axis – top to bottom).

d. One-dimensional Median Filter

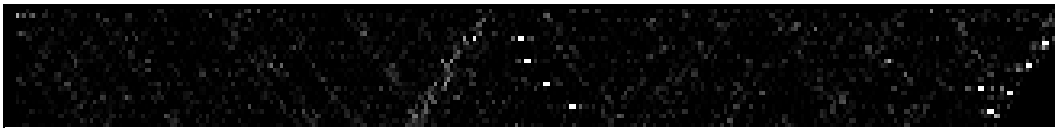
At that point, the one-dimensional median filter is applied along the time axis of the signal images to smooth the TCF phase output information prior to the differentiation operation, as described in Chapter IV.A. A separate function was written to scan the image down the desired axis using the MATLAB function *medfilt1.m.* Figure 34 shows the results of the one-dimensional median filter on the TCF phase output image shown in Figure 33.

Figure 34. 1-D Median filtered output of TCF phase output in Figure 33 (Horizontal axis: Time axis – left to right, Vertical axis: Lag axis – top to bottom).

e. Derivative Filter

Two methods were investigated, as previously discussed in Chapter IV. First, the time axis difference was conducted using a 2^{nd} order difference filter from the MATLAB function *diff.m*. Second, a differentiation filter known as the Savitzky-Golay (SG) Finite Impulse Response (FIR) smoothing filter was selected as this filter removes noise while performing the differentiation operation. This differentiation operation was implemented in the function *diffsg.m* with the MATLAB function *sgolay.m*, and is included in Appendix A. While both the difference filter and SG filter are shown in Figure 35 for illustration purposes, only SG and segmentation option B are displayed for the remainder of these examples.

a.



b.

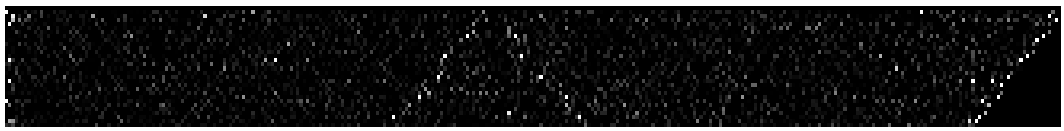


Figure 35. Figure 34 filtered by: a. 1st derivative, 9th order SG filter down time axis using segmentation option B, b. 2nd order difference filter down time axis using segmentation option A (Horizontal axis: Time axis – left to right, Vertical axis: Lag axis – top to bottom)

f. Rounding Operation

The 9th order SG filter implementation of the 1st order derivative emphasized minor amounts of noise in images that needed removal as the noise level of the QPSK signals increased. As a result, the MATLAB *round.m* function was applied to prepare these remaining noise pixel values for removal by the two-dimensional 3x3 mean filter prior to its application. As mentioned in Chapter IV, noise overwhelmed the edge detection process in low SNR levels when the two-dimensional mean filter was not applied, while at the same time a high-order two-dimensional mean filter resulted in erasing phase shift timing information. Thus, the rounding operation provided a simple way to keep the smaller two-dimensional mean filter performing adequate noise removal while ensuring phase shifts were not removed as well. Figure 36 provides the results obtained after the rounding operation applied to Figure 35.

Figure 36. Rounding of SG results of Figure 35 (Horizontal axis: Time axis – left to right, Vertical axis: Lag axis – top to bottom).

g. Two-dimensional Mean Filter

The 3x3 two-dimensional mean (Chapter IV, Section B.3) filter removed noise left by the differentiation step, whereas phase shift information was not erased. Figure 37 shows the two-dimensional mean filter of image in Figure 36.

Figure 37. Two-dimensional 3x3 mean filter results of Figure 36 (Horizontal axis: Time axis – left to right, Vertical axis: Lag axis – top to bottom).

h. Roberts Edge Detection

The Roberts edge detection method in the MATLAB function *edge.m* was used to provide this edge enhancement operation. Figure 39 shows the Roberts edge detection operation on the two-dimensional mean filtered output of Figure 37.

Figure 38. Roberts edge detection on two-dimensional 3x3 mean filter of Figure 37 (Horizontal axis: Time axis – left to right, Vertical axis: Lag axis – top to bottom).

i. Morphological Processing

MATLAB functions *imdilate.m* and *imerode.m* were used to perform the morphological operations during the image processing phase. Details of these operations can be found in Chapter IV Section D. The final image before masking is shown in Figure 39.

Figure 39. Results of final erosion on Figure 38 (Horizontal axis: Time axis – left to right, Vertical axis: Lag axis – top to bottom).

j. Image Masking Operation

Simulations showed that a boundary condition problem remained no matter which segmentation option was used. However, this problem was decreased by selecting segmentation option B. The TCF computation boundaries result in edges oriented in the same direction as the phase shift characteristic. Thus, the leading and trailing 32 samples of signal information are removed from the image to eliminate these edge boundaries, which do not carry phase shift time information. Masking was conducted by replacing the last 32x30 pixel region of each signal window with zeros, resulting in removing TCF created edges contained within this region from consideration. Figure 40 shows the final processed output image after performing the masking operation on Figure 39.

Figure 40. Final image - results of masking operation on Figure 39 (Horizontal axis: Time axis – left to right, Vertical axis: Lag axis – top to bottom).

B. 2-D MATCHED FILTER IMPLEMENTATION

1. Matched Filter Introduction

The matched filter correlates a noisy signal with a desired signal characteristic to track the occurrence of that characteristic in time or position. Our two-dimensional matched filter uses images to track phase shift occurrences contained in the processed TCF phase matrix. This process was implemented using SIMULINK in MATLAB and specifically made use of the *2-D Correlation* block in the *Video and Image Processing* blockset. The 2-dimensional correlation output between the processed TCF phase image A of size *Ma*x*Na* and the target image B of size *Mb*x*Nb* is given as:

$$C(i,j) = \sum_{m=0}^{(Ma-1)} \sum_{n=0}^{(Na-1)} A(m,n) \bullet conj(B(m+i, n+j)), \tag{5.1}$$

where the size of matrix A is larger than that of matrix B and $0 \le i < Ma + Mb - 1$ and $0 \le j < Ma + Mb - 1$.

Note that the default normalization option of the *2-D Correlation* block gave poor correlation values when dealing with noisy signals. Thus, we implemented a different normalization factor to give maximum cross-correlation values when the target object fit inside the triangular shape in the processed TCF phase image and is given by:

$$K_{norm} = \frac{1}{\sum_{i=0}^{I-1}\sum_{j=0}^{J-1} B(i, j)}, \tag{5.2}$$

where B is the target image of size MbxNb.

2. SIMULINK Implementation

Figure 41 presents the SIMULINK model, *patmatch.mdl,* designed to extract phase shift time information. The figure shows:

- One signal image block, *Signal A,* which contains the processed and masked TCF phase image obtained after filtering and morphological operations, of size 332*30,
- Target image blocks, *Tgt Image A* and *Tgt Image B,* shown earlier in Figure 28, and their corresponding normalization factors, *Norm A* and *Norm B,* described in Equation (5.2),
- Two *2-D XCORR* blocks which perform the two-dimensional cross-correlation operations in parallel,
- Two max location blocks which compute the index for the location of the maximum correlation value,
- The *Window Time Correction* block corrects for: 1) the *2D XCORR* block reference frame offset from the center of the target triangle along the time axis and 2) for the 32 pixels cropped from the top of the signal image (time axis).

- The *Noshift Time Correction* block simplifies programming when a no-shift decision has been made.

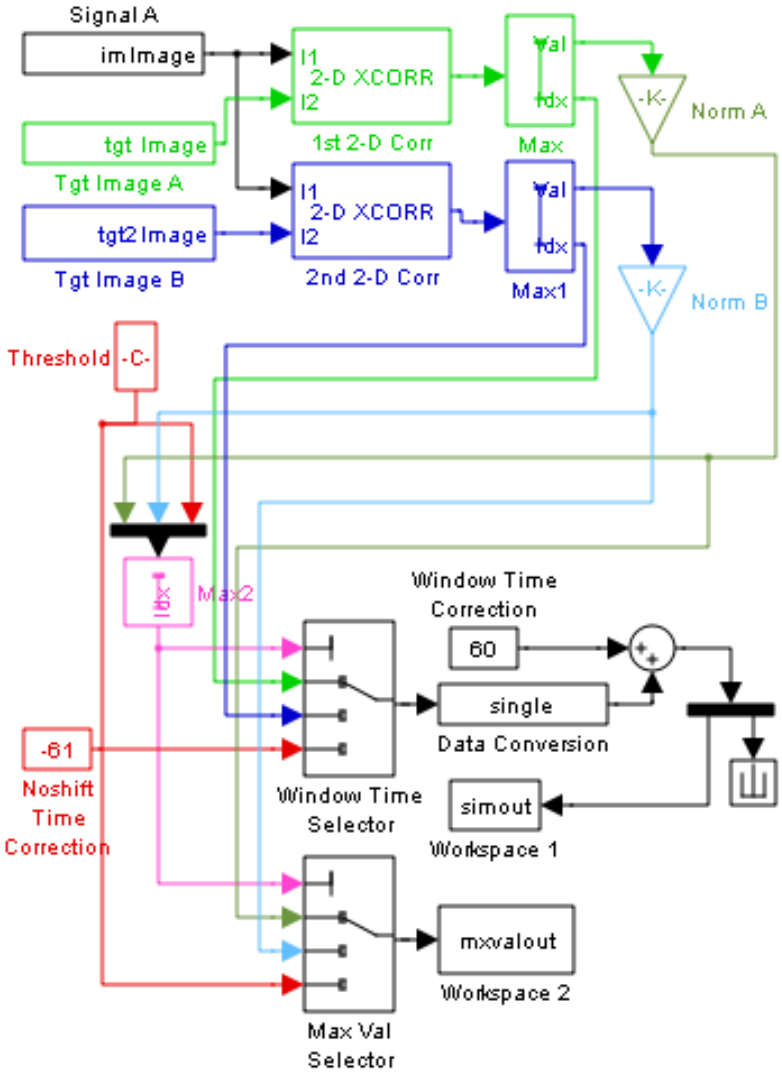


Figure 41. 2-D Matched Filter SIMULINK implementation, *patmatch.mdl*.

3. Patmatch.mdl Model Results

Figure 42 illustrates results obtained using a processed and cropped TCF phase image both for one-shift (Figure 42.a) and no-shift cases (Figure 42.b) for QPSK signals with 6dB SNR levels. In both cases, the combination of image cropping and internal

program processing give the resulting TCF phase images dimensions of 328 pixels long by 30 pixels wide in preparation for the matched filter operation. Note how the phase shift stands out following the image pre-processing steps while in the no-shift case image noise is completely erased. This visual separation of the phase shift and no-shift case permits the use of a two-dimensional matched filter approach in extracting phase shift times.

Figure 42. Processed and Cropped TCF Phase images from PSK signal + AWGN, SNR=6dB; a. one-shift located at time 150, b. no-shift (Horizontal axis: Time axis – left to right, Vertical axis: Lag axis – top to bottom).

In this matched filter scheme, the desired phase shift information is contained in the white triangular object shown in Figure 42.a. The matched filter uses the target image to "scan" processed TCF phase images obtained from noisy PSK signals. By using appropriate target images, such as those shown in Figure 28, and repeated in Figure 43, cross-correlation output values are expected to be high when a phase shift is present at the shift time location and low when there is no phase shift in the analysis window.

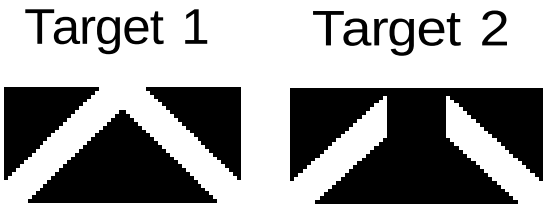


Figure 43. Target Images of size 60x30; same as Figure 28 (Horizontal axis: Time axis – left to right, Vertical axis: Lag axis – top to bottom).

Figures 44 and 45 show the two-dimensional cross-correlation output obtained for the two target images defined in Figure 43 applied to images contained in Figure 42. The

2-D XCORR block stops scanning 60 pixels before the end of the signal image due to the size of the target images, resulting in 268 possible correlation values corresponding to scan positions, as shown in Figure 44. This scan results in the maximum correlation value located at position index 90 along the time axis. The phase shift is found at location 152, after taking into account the 30 pixel offset due to the target image length and 32 pixel offset resulting from the cropping stage. Figure 45 shows that the cross-correlation values are equal to 0 for the no phase shift case, as the processed TCF phase image was completely blank. Next, we discuss the two-dimensional matched filter threshold determination.

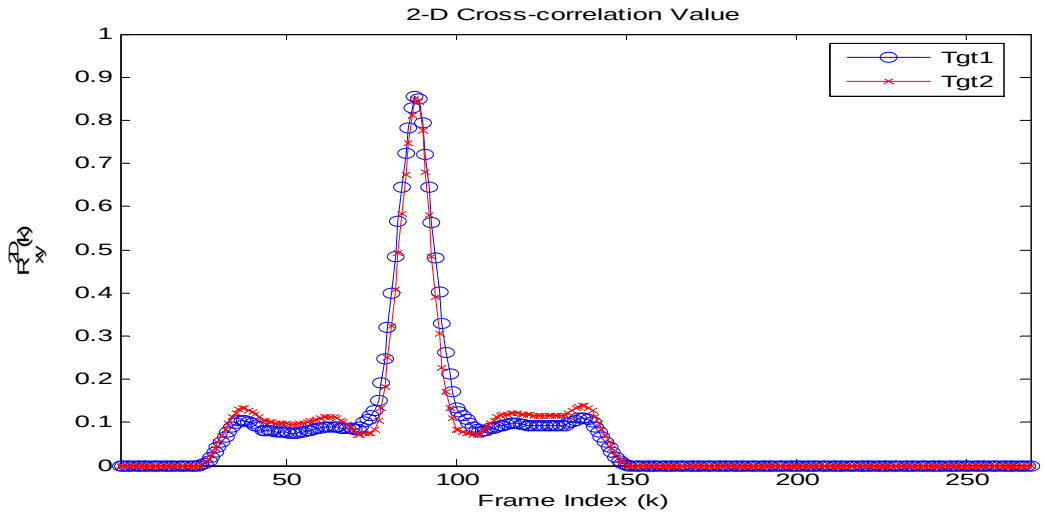


Figure 44. Output of 2-D Cross-correlation blocks for target images 1 and 2 and Figure 42.a phase shift input image.

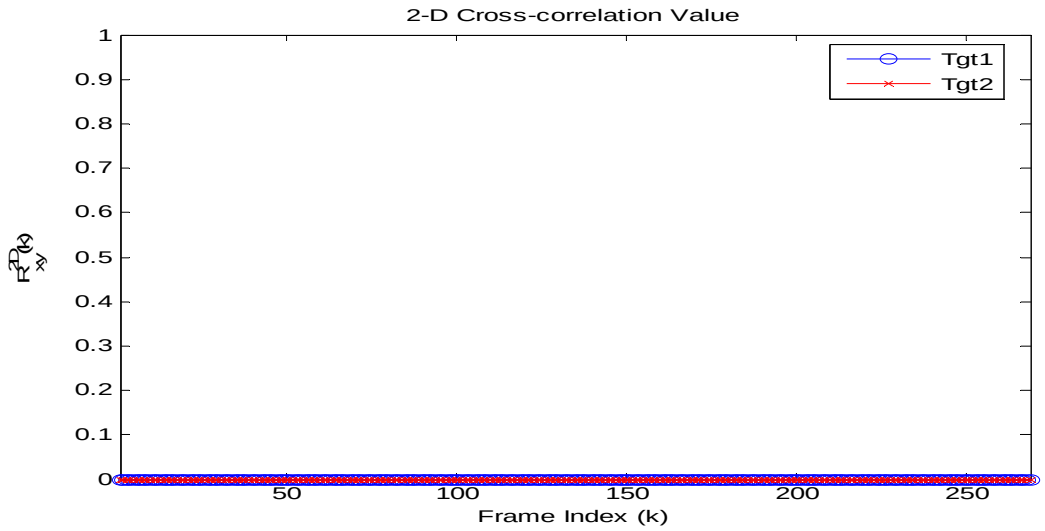


Figure 45. Output of 2-D Cross-correlation Blocks for target images 1 and 2 and Figure 42.b no-shift input image.

C. THRESHOLD DETERMINATION

The matched filter output provides correlation values between target and processed TCF phase images. These correlation values characterize the quality of the match between these objects within the images and indicate where the best match occurs in the TCF phase image. Thus, the next step lies in the identification of a threshold value above which the user decides the match is high enough to say there is a phase shift within the analysis frame and identify its specific time location. The detection algorithm uses a threshold value determined experimentally for a range of SNR levels. The normalized cross-correlation range between zero and one requires that our image processing maximize the difference between the phase shift and no-shift cases. Histogram plots of the maximum cross-correlation values for successive phase shift and no-shift trials were used to find specific threshold gaps in the resulting correlation value groupings. Noise SNR levels of 12, 9, 6, 4, 2, and 0 dB were investigated with tests of 500 phase shift and 500 no-shift scenarios where the maximum correlation values were recorded separately for each.

A few comments can be made on results observed:

- Simulations showed that the threshold value could easily be determined for signals with SNR levels down to 6dB or higher. Figure 46 plots results obtained for a 9dB SNR case. Down to that 6dB SNR level, filtering and morphological operations applied to the TCF phase image clean the image well, resulting in the non-overlapping cross-correlation values observed for no-shift and one-shift signal cases. Simulations showed that a cross-correlation value threshold equal to 0.28 worked well down to 6dB. Appendix B includes results obtained for additional SNR values.

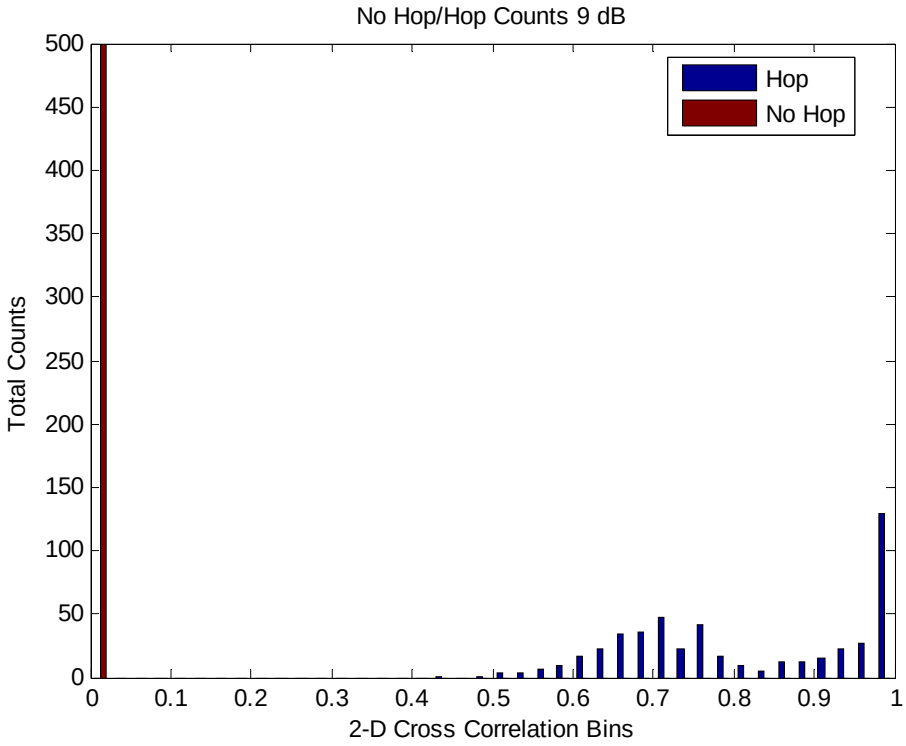


Figure 46. Cross-correlation values for QPSK signal + AWGN, SNR=9dB, phase shift and no-shift cases.

- Simulations showed that results degrade quickly when the SNR level falls below 4dB. Figure 47 shows cross-correlation values obtained for one-shift and no-shift signal cases for PSK signals at 4 dB SNR. At that level, we selected a threshold value equal to 0.45 to limit the number of false alarms and maximize the number of phase shifts detected.

- Additional simulations were used to obtain updated threshold values when the SG filter was selected for the differentiation step. Note, the algorithm was not optimized for the SG filter or its noise removal capabilities, resulting in the compressed range of threshold values listed in Table 1.

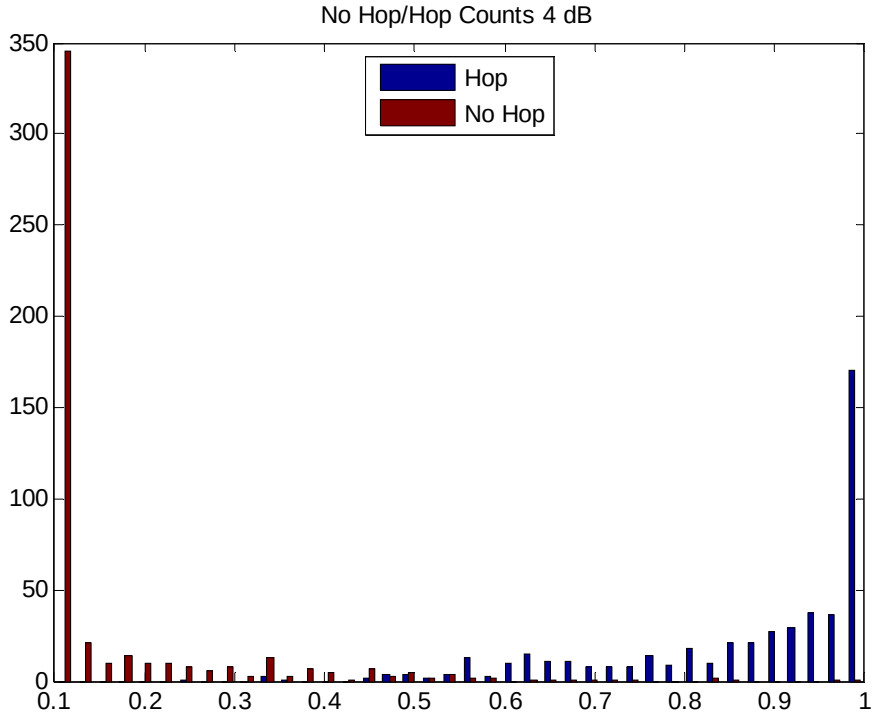


Figure 47. Cross-correlation values for QPSK signal at 4 dB SNR, phase shift and no-shift cases.

SNR (dB)	**12**	**9**	**6**	**4**	**2**	**0**	**-2**
Difference filter	0.28	0.28	0.28	0.45	0.5	0.8	0.98
SG filter	0.42	0.42	0.2	0.2	0.1	0.01	0.01

Table 1. Difference filter and SG filter Simulation Threshold Values per SNR

D. OVERLAPPING WINDOW IMPLEMENTATION

In attempting to improve detection performances in noisy environments, we investigated a window overlapping scheme designed to provide multiple looks of the same phase shift timing information, as illustrated in Figure 48. The overlap method allows the user to first reach a separate phase shift decision in each analysis frame and then combine decisions, as appropriate. This section discusses the decision method followed when dealing with overlapping windows. Simulations showed the overlapping window implementation helped in cases where phase shifts appeared in edges in the no overlap case and in cases where noise levels were high causing many false alarms.

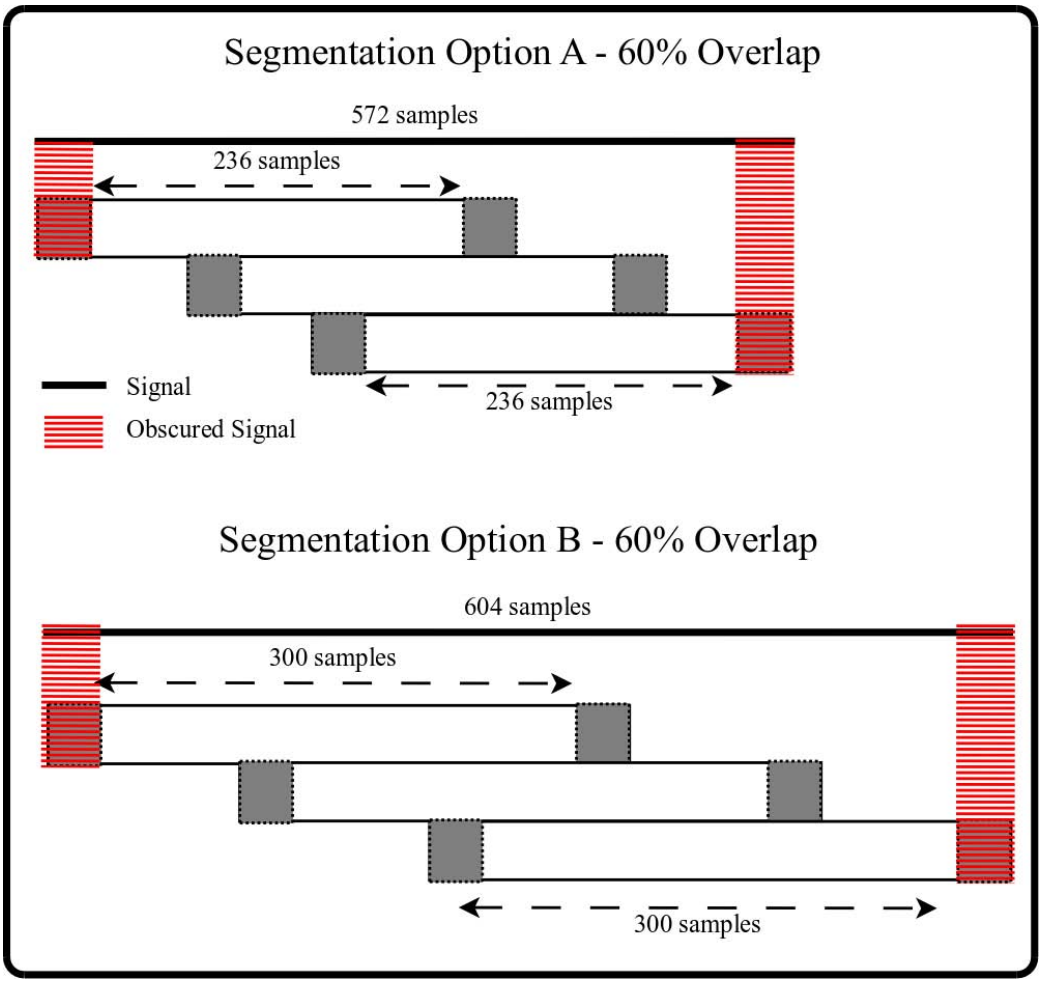


Figure 48. Signal Segmentation for 60% Overlap.

Figure 49 shows the general program flow for the window overlapping method. Two decision factors were defined to adjust the sensitivity of the scheme. First, the No-shift factor (Z) controls the maximum number of allowed individual no-shift decisions in a group of overlapping windows. Second, the Decision (D) factor specifies the minimum number of overlapping windows that must agree, up to a tolerance level, for a phase shift time to be confirmed.

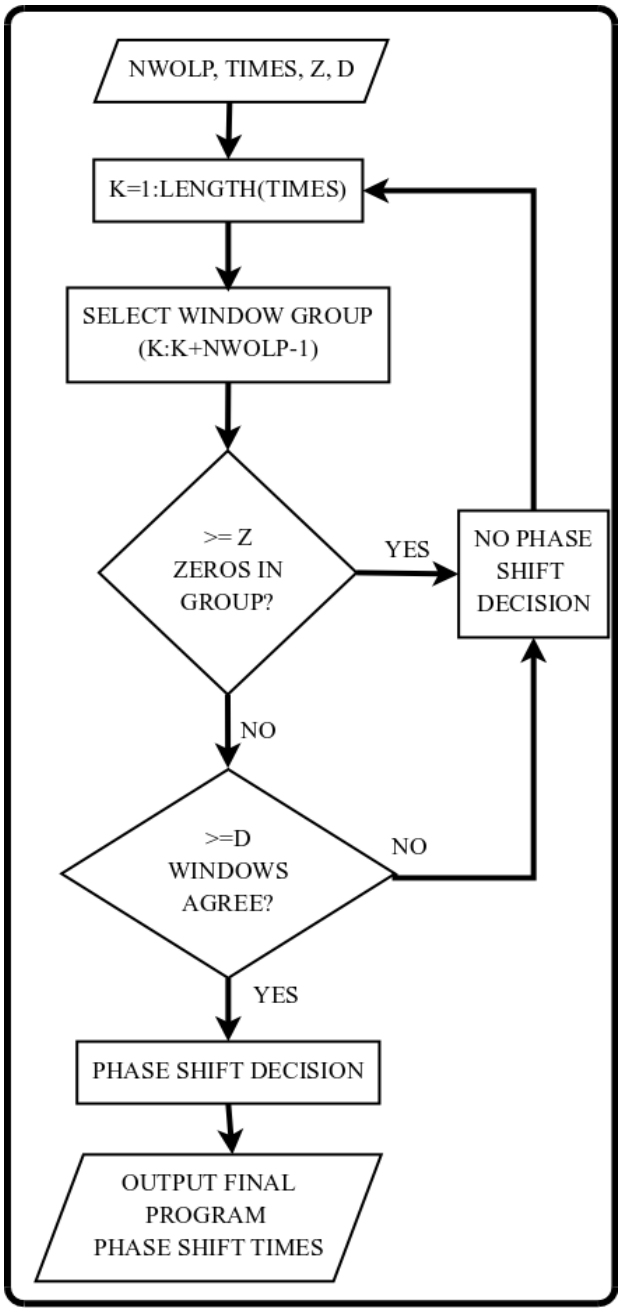


Figure 49. Window Crosscheck Function Flow.

Table 2 summarizes the various options for varying overlap cases. "Group size" refers to the number of overlapping windows obtained for a specific overlap rate. Note that the no-shift factor and decision factor do not apply for the zero overlap case, as final decisions are based on one individual decision only.

Overlap (%)	Group Size	Zero Factor	Decision Factor
0	1	NA	NA
50	2	1	2
60	3	2	2
80	5	3	3
90	10	5	4

Table 2. Window Verification Factors based on Overlap.

First, individual phase shift decisions are made, based on the decision threshold value alone, and the accompanying phase shift time or a zero, when a no-shift decision is reached, are stored in the TIMES vector. Next, a specific number of windows represented here as "Group Size," and directly linked to the overlap rate are examined as a group as follows (Figure 50 illustrates this process):

- If the group of overlapping windows contain greater than or equal to Z numbers of zero (no-shift) decisions, the TIMES index is incremented by one, resulting in a no-shift decision for that group. Otherwise the program compares individual window groups, as follows.
- When fewer than Z windows return a no-shift decision, and greater than or equal to D windows agree that a phase shift exists within a user-specified time range, as defined by the COMP parameter, the "best" shift time is chosen and stored. For example, estimated shift times found within 30 samples of each other are taken into account for final phase shift decisions for a window length of 300 samples and a 10% tolerance level. At that point, the final phase shift time is obtained from the group of potential values following the process illustrated in Figure 50.
- When fewer than Z windows return a no-shift decision, and fewer than D windows agree that a phase shift exists within a specific time range, the

TIMES index is incremented by one, resulting in a global no-shift decision for that group. This process continues until all individual windows have been scanned.

For example, a specific signal time location can be observed by three consecutive windows for the 60% window overlapping case. Parameters are set such that there can be no more than one no-shift in the group and two of the windows lead to estimated phase shift times differing by less than a user specified amount for an overall phase shift decision to be reached. The global phase shift decision process derived for the overlapping window case is further described by the flow chart included in Figure 49 and illustrated with the example contained in Figure 50.

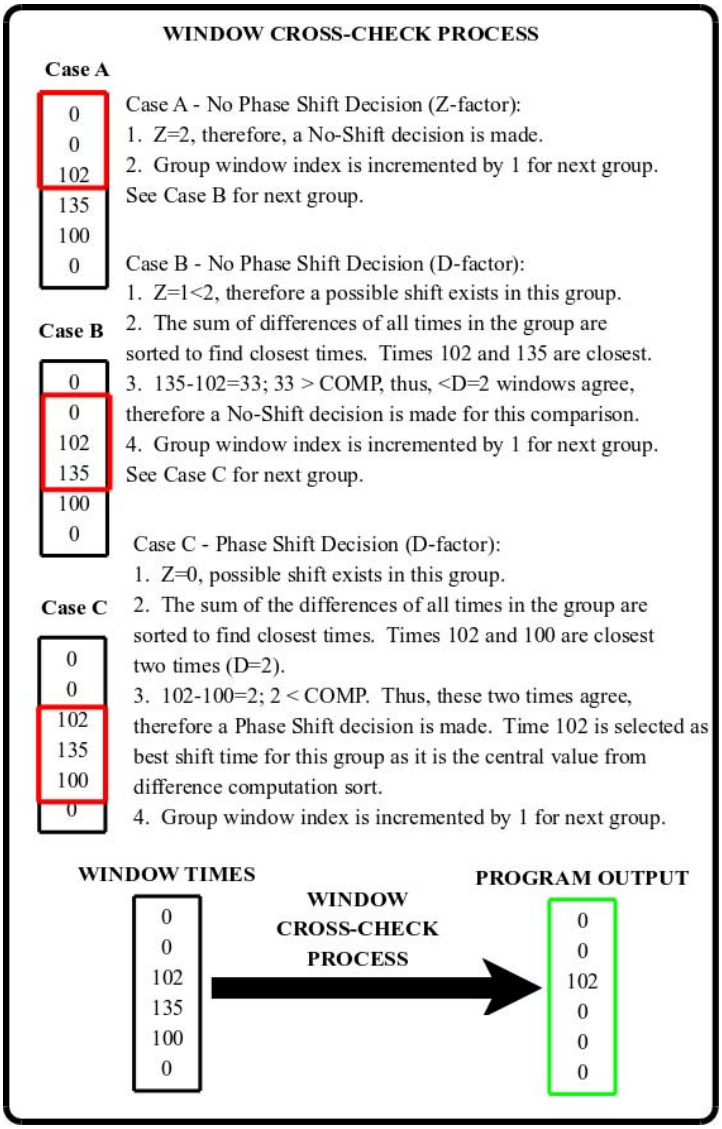


Figure 50. Phase shift Time Selection Process, 60% Overlap, 10% Tolerance, Frame Size 300.

E. CONCLUSIONS

This chapter provided an overview of the phase shift detection algorithm, described the two-dimensional matched filter implementation, threshold determination and a window overlapping algorithm implemented to identify phase shift timing found in noisy QPSK signals. MATLAB files and data plots not included in this chapter are included in Appendices A and B. The next chapter presents results obtained.

THIS PAGE INTENTIONALLY LEFT BLANK

VI. TESTING AND SIMULATION RESULTS

This section presents detection performance results obtained for the PSK phase shift time identification scheme considered in this study. The implementation involves the selection of several user-defined parameters and images such as the analysis frame length and overlap amount, structuring element shapes and sizes, target images used for the matched filter implementation, and threshold values for the phase shift versus no-shift decision. As a result, various combinations of parameter values were considered during the study in an effort to estimate a "best" set of parameters, where "best" was evaluated in terms of resulting probability of accurate detection of the phase shift time information, probability of accuracy, and probability of false alarm.

The overall phase shift timing identification algorithm involves a differentiation step, as shown earlier in Figure 35. Chapter IV discussed two different implementations of this differentiation step: a basic differentiation filter, and a noise robust implementation using the SG differentiation filter. This section presents results obtained using both differentiation operators. First, we present identification performances derived with the difference filter differentiation step. Specific structuring elements shapes and sizes, and threshold values were investigated and the "best" combinations selected for that specific differentiation operator. Next, we present performance results obtained when the SG differentiation filter is selected in the differentiation step.

A. SIMULATION OVERVIEW

We generated QPSK signals distorted by AWGN in SNR levels between 12 and -2 dB. Testing was accomplished using 500 non-overlapping windows per experiment. Two-hundred fifty windows contained either a phase shift or no-shifts, with shifts spread semi-randomly throughout the length of the signal when window overlapping was investigated. The no-overlap case did not use random placement to ensure only one shift or no shift per window. The scenarios used window lengths of 256 and 300 to investigate window length impacts on the resulting detection algorithm. Windows of length 300 were used for all other testing.

Performance results are presented in terms of the probability of correct detects (PCD), false alarms, accuracy (P_{ACC}), and error rates (P_{ERR}). The probability of correct detection is defined as the probability of deciding there is a phase shift in the signal and its estimated location is within a given range specified by the COMP parameter introduced in Chapter V. The probability of false alarm is defined as the probability of detecting a phase shift when there is no actual phase shift. P_{ACC} and P_{ERR} are defined below in Equation(6.1):

$$P_{ACC} = P(0 \mid 0) + P(1 \mid 1) \text{ and } P_{ERR} = P(1 \mid 0) + P(0 \mid 1), \tag{6.1}$$

where respective probabilities were defined as follows:

$$\begin{aligned} &P(0 \mid 0) = \text{Probability of correctly detecting no phase shift,} \\ &P(1 \mid 1) = \text{Probability of correctly detecting a phase shift,} \\ &P(1 \mid 0) = \text{Probability of detecting a phase shift when no shift is present, and} \\ &P(0 \mid 1) = \text{Probability of deciding no phase shift when there is a shift present.} \end{aligned} \tag{6.2}$$

Probability quantities derived in this section are expressed on a 0 to 1 scale, where 1 represents 100%. This program was optimized for the difference differentiation step and an expected signal noise level of 6 dB SNR, however, final tests considered noise levels ranging from -2 dB to 12 dB SNR levels.

B. DIFFERENCE FILTER OPTION RESULTS

1. Difference Filter Simulation Settings

Table 3 below shows settings for the difference filter algorithm test runs. Segmentation B runs use an edge offset of 32 samples appended to each end of the desired frame size.

Threshold	Analysis window Overlap amount (%)	Phase Shift time Accuracy Tolerance (% of window length)	Frame Size (n)
See Table 1.	0, 60, 90	5, 10, 15	256, 300

Table 3. Difference filter simulation parameter settings using segmentation options A and B.

2. Segmentation Results

The Segmentation B method described earlier in Section V.A.4.a, insured that no sections of the signal were missed during the phase shift identification process. Figures 51 and 52 present the probability of correct detection obtained for the no-overlap, 60% and 90% window overlap implementations. Figures 53 and 54 present the probability of false alarms obtained for the same configurations. A few comments can be made:

- Results show no significant difference for the probability of correct detection and the probability of false alarms between the two segmentation methods down to 4dB SNR and both degrade significantly below that SNR level. This similarity is to be expected as all shifts are inserted around window centers for the no-overlap implementation to prevent two shifts from being located in a single window, and to avoid locating shifts close to window edges, where they would be masked during pre-processing steps. Note that no such shift placement constraint is imposed for the window overlap case, resulting in phase shifts potentially located towards edges of the analysis window where they may be masked and result in missed phase shifts, resulting in degraded performances.
- Results show that a higher overlap percentage results in a slightly higher probability of correct detection and lower probability of false alarms down to 4dB SNR.
- False alarm results obtained for SNR levels below 4dB show the identification is no longer useable for the set of user-specified parameters selected in this work.

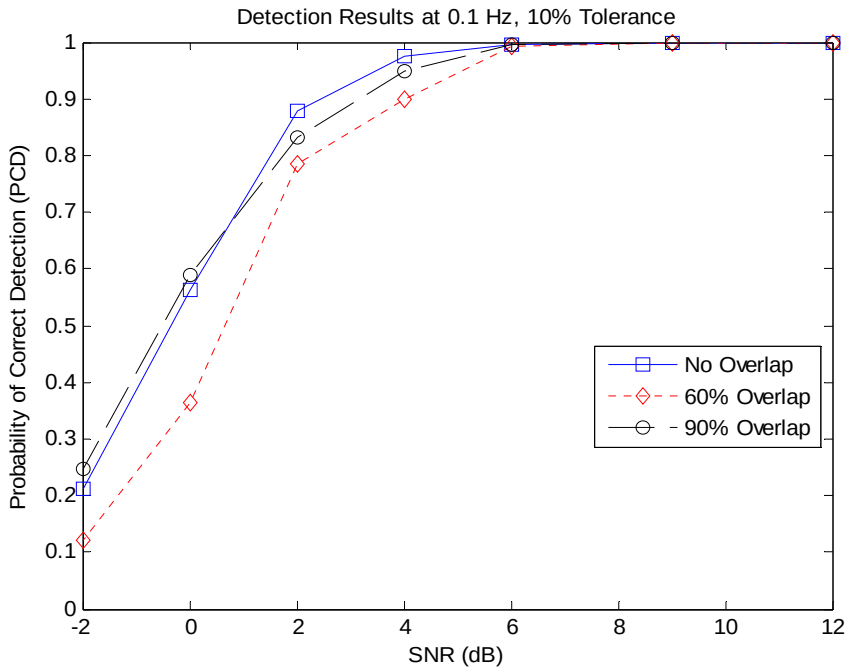


Figure 51. Segmentation A Probability of Correct Detection, signal carrier frequency = 0.1Hz, 10% accuracy detection tolerance.

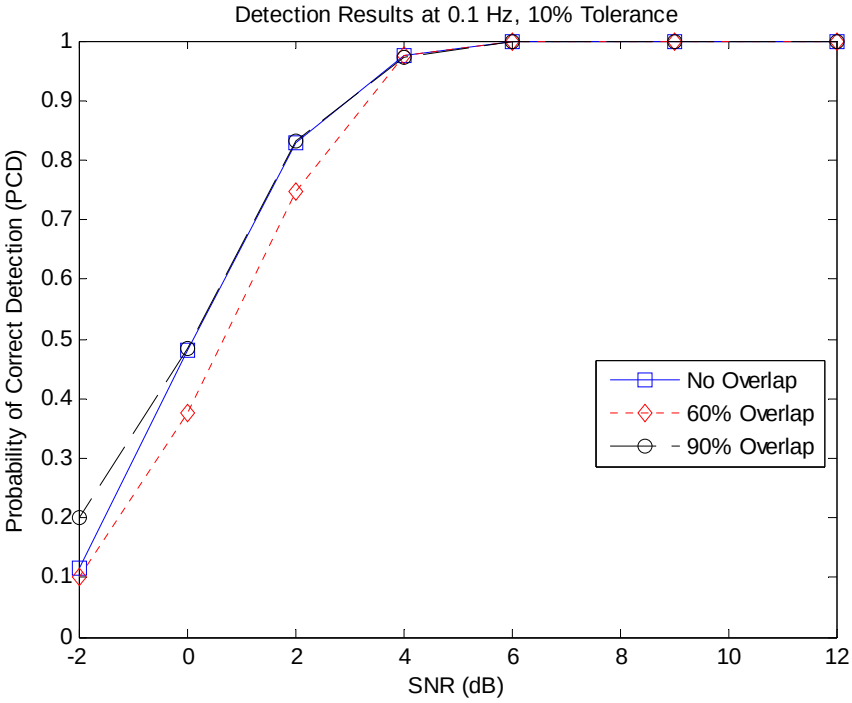


Figure 52. Segmentation B Probability of Correct Detection, signal carrier frequency = 0.1Hz, 10% accuracy detection tolerance.

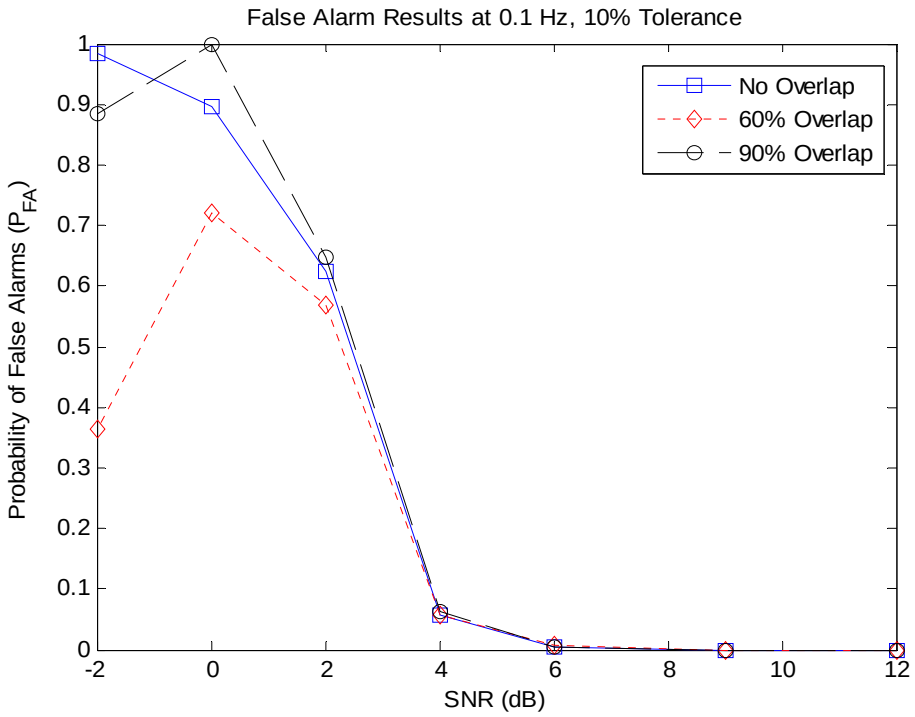


Figure 53. Segmentation A Probability of False Alarms, signal carrier frequency = 0.1Hz, 10% accuracy detection tolerance.

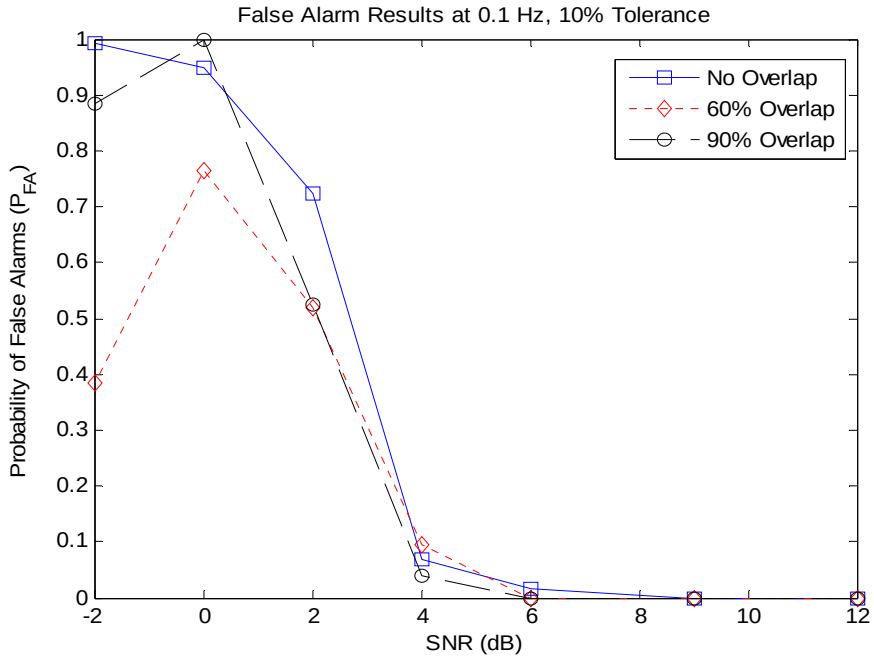


Figure 54. Segmentation B Probability of False Alarms, signal carrier frequency = 0.1Hz, 10% accuracy detection tolerance.

Figures 55 and 56 present the probability of accuracy, which takes into account both accurate shift and no-shift detections, for both segmentation schemes, while Figures 57 and 58 show the probability of error. These two probability quantities provide an overall metric on the success of each method. Accuracy and error plots show similar trends for the three overlap cases considered down to 4dB SNR. Again, performances degrade significantly below that SNR level.

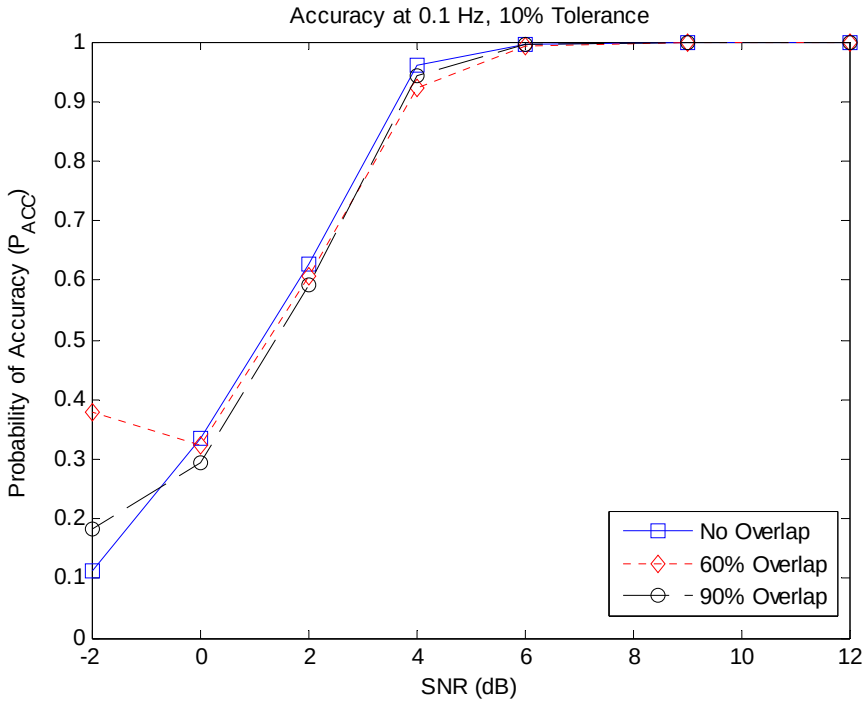


Figure 55. Segmentation A probability of accuracy, signal carrier frequency = 0.1Hz, 10% accuracy detection tolerance.

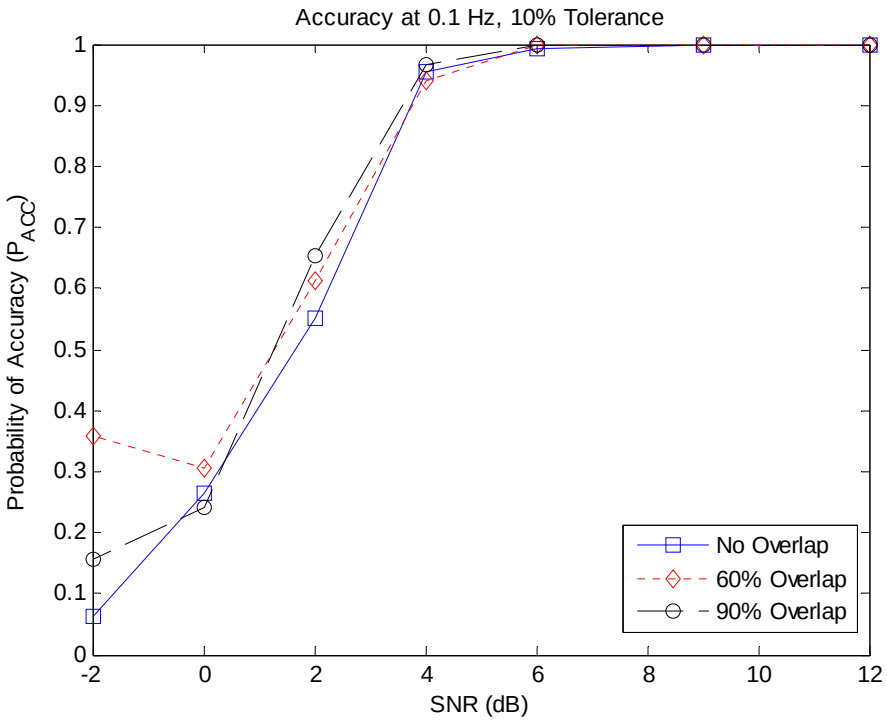


Figure 56. Segmentation B probability of accuracy, signal carrier frequency = 0.1Hz, 10% accuracy detection tolerance.

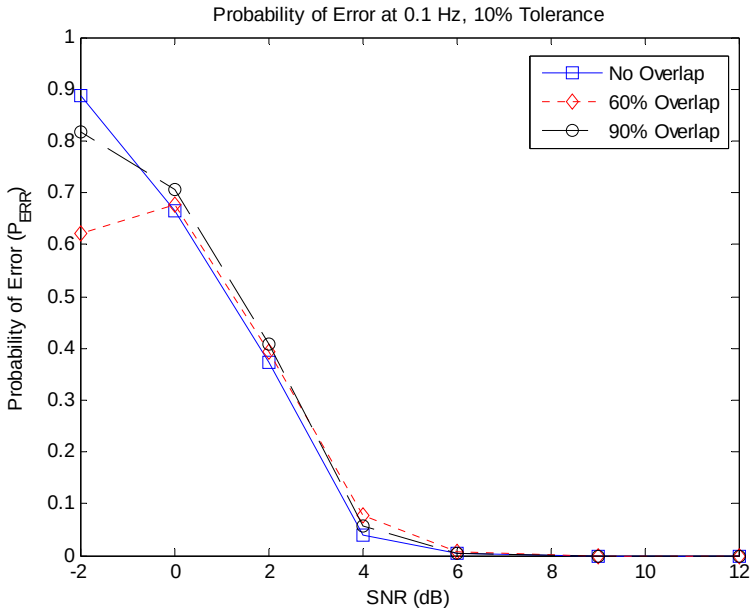


Figure 57. Segmentation A Probability of error, signal carrier frequency = 0.1Hz, 10% accuracy detection tolerance.

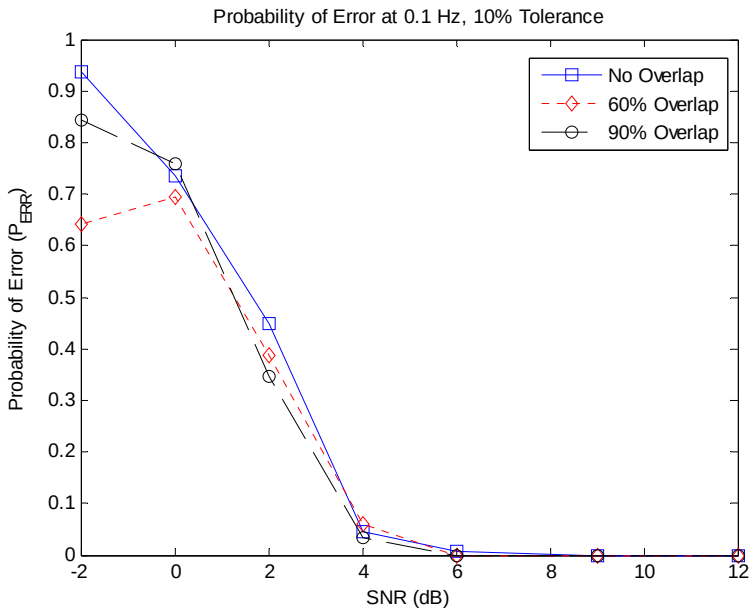


Figure 58. Segmentation B Probability of error, signal carrier frequency = 0.1Hz, 10% accuracy detection tolerance.

The remainder of the chapter presents performance results obtained for the Segmentation B option only, as it is designed to cover the entire signal length.

3. Overlap Results

Simulations showed that overlapping analysis windows to provide multiple looks of the same time ranges was beneficial for SNR levels between 2 and 4dB as it reduced the probability of error and increased the probability of correct detection (see Figures 52 to 58). Results also show perfect detection and null false alarm rate for SNR levels above 6dB.

Note that the 90% window overlap configuration leads to a higher rate of false alarms than the 60% configuration does. This result is due to the decision rule implemented in the study, and a different overall decision rate may be obtained by changing it.

4. Window Length Results

Two different window lengths were considered in the study to evaluate window length impact on resulting performances; windows of lengths 256 and 300. Figures 59 and 60 present performance results obtained for the segmentation B option, 10% detection accuracy tolerance, signal carrier frequency at 0.1 Hz, no window overlap and 60% overlap rates. A few comments can be made:

Results show that overall accuracy and error rates obtained for 0% overlap and 60% overlap are similar for both window lengths at a given overlap rate down to 4dB SNR levels. Again, performances degrade significantly below that level. These results are to be expected, as no section of the signal is lost during the pre-processing phase.

Window lengths equal to 300 samples are used for the remainder of testing.

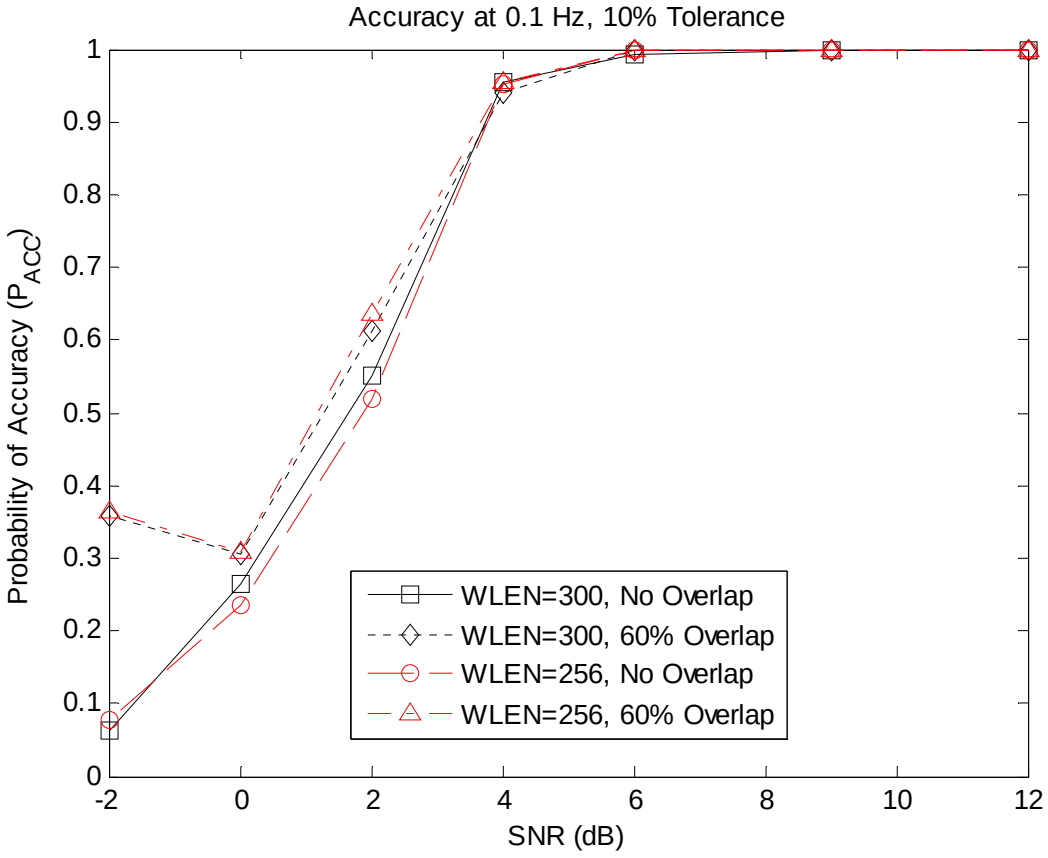


Figure 59. Difference filter probability of accuracy with window lengths of 256 and 300 samples at 0% and 60% overlap.

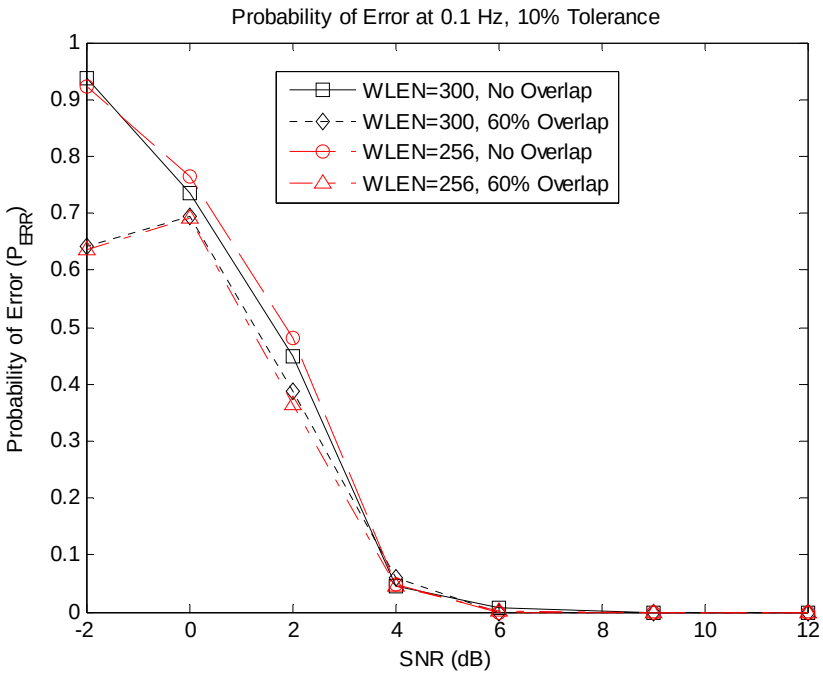


Figure 60. Difference filter probability of error with window lengths of 256 and 300 samples at 0% and 60% overlap.

5. Frequency Results

Next, we investigated the impact of signal carrier frequency on resulting performances by comparing results obtained for a low (0.1Hz) and high (0.4Hz) normalized carrier frequencies. Note that signal frequency information is contained in the lines of the TCF phase output images. However, such information got completely removed by the differentiation step for noise-free signals. Therefore, carrier frequency variation was expected to have little impact on performances in medium to high SNR levels. Figures 61 and 62 plot the probability of accuracy and error for the two signal carrier frequencies considered, no-overlap and 60% overlap configurations, and 10% tolerance levels. Results show similar performances are obtained for both carrier frequencies considered down to 4dB. Performance becomes different for SNR levels below 4dB, likely a result of the TCF phase information degradations resulting from noise distortions.

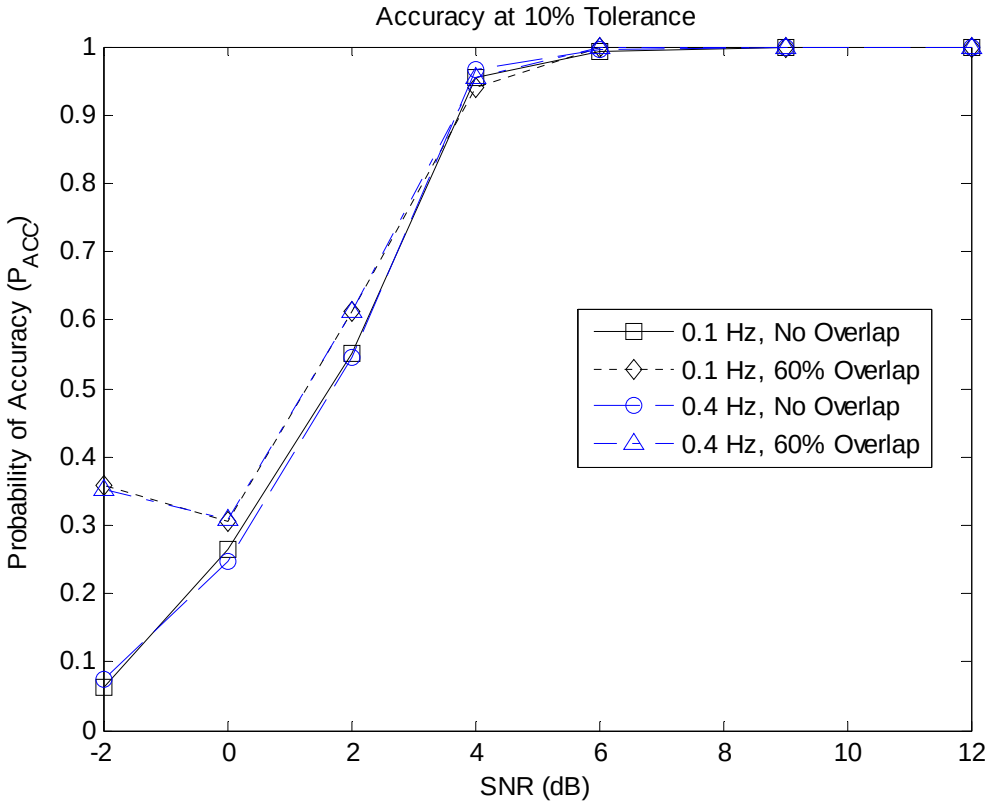


Figure 61. Difference filter algorithm probability of accuracy, normalized signal carrier frequency = 0.1 Hz, 0.4 Hz, no-overlap and 60% window overlap, 10% accuracy detection tolerance.

Further results consider the signal normalized frequency equal to 0.1 Hz only.

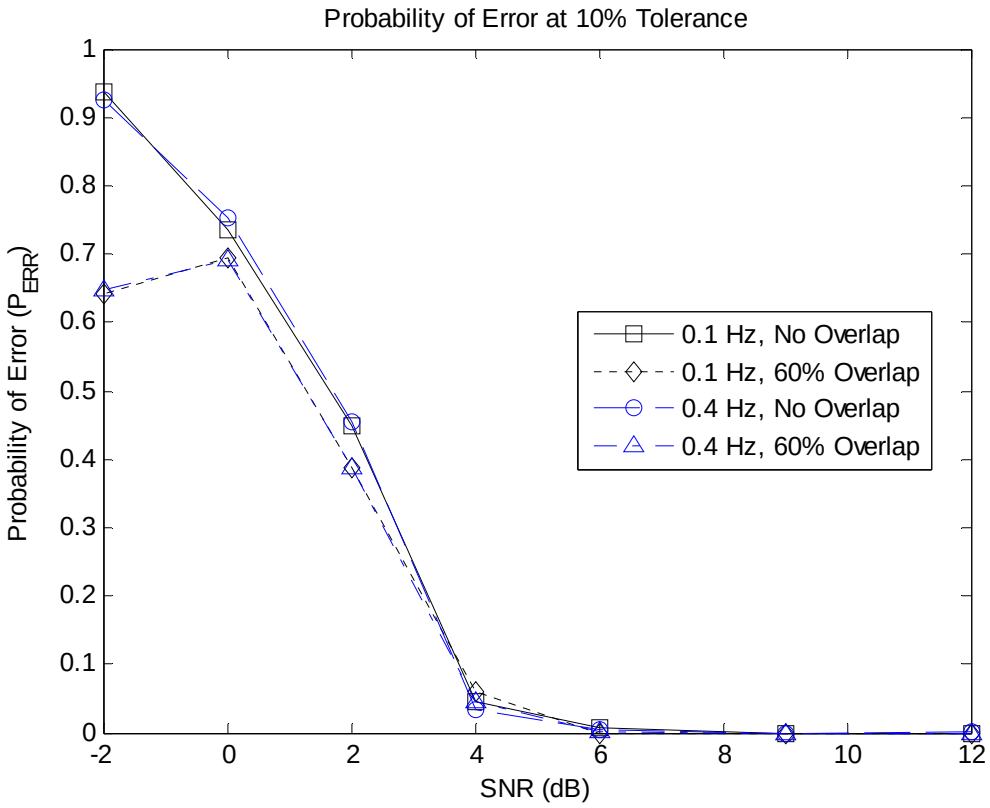


Figure 62. Difference filter algorithm probability of error, normalized signal carrier frequency = 0.1 Hz, 0.4 Hz, no-overlap and 60% window overlap, 10% accuracy detection tolerance.

6. Tolerance Results

The detection accuracy tolerance parameter is one of the more sensitive user-specified parameters, as it specifies how much error is to be allowed between true and estimated phase shift locations. All initial simulations conducted to select parameters needed during the pre-processing phase such as morphological operators, filter orders, etc… used a 10% tolerance level. Next, additional tolerance levels equal to of 5%, 10%, and 15% were considered using these parameters.

Recall that the parameter COMP is the maximum allowable phase shift time deviation for a shift decision and is based on tolerance level and window size as set by

the user. Thus, a 15% tolerance case permits ±45 samples around the actual phase shift time to be considered a valid shift for a 300-sample window length.

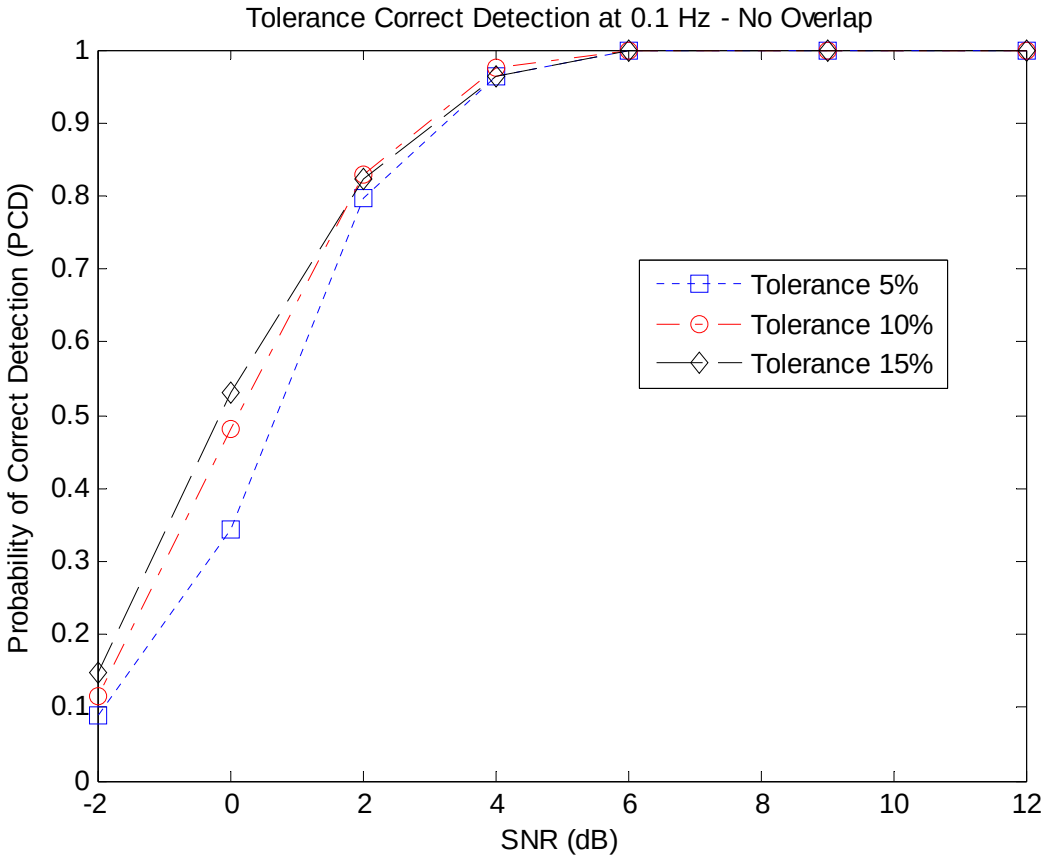


Figure 63. Difference filter algorithm PCD, signal normalized frequency = 0.1 Hz, no window overlap, 5%, 10%, and 15% accuracy detection tolerance.

Figure 63 and 64 present the probabilities of correct detection and false alarms obtained for 5, 10 and 15% tolerance levels for a normalized signal carrier frequency equal to 0.1Hz in SNR levels between -2 and 12dB. Results show the probability of correct detection is the same for SNR levels above 4 dB for the three tolerance levels investigated. Results also show detection performances degrade as the tolerance level decreases for SNR levels below 4dB. False alarm rates show similar performances for the three tolerance levels considered.

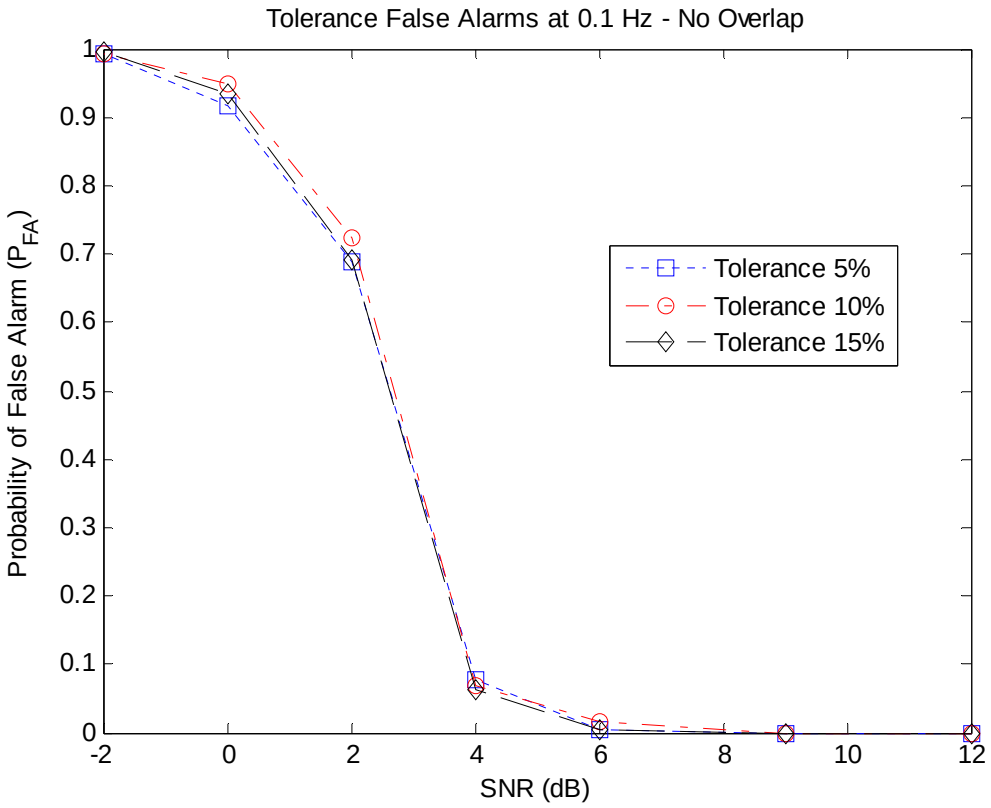


Figure 64. Difference filter algorithm probability of false alarm, signal normalized frequency = 0.1 Hz, no window overlap, 5%, 10%, and 15% accuracy detection tolerance.

This section discussed performances obtained when the difference filter algorithm is used in the differentiation step, compared segmentation options, and discussed the algorithm sensitivity to overlap, signal carrier frequency, window length, and tolerance levels. Recall that one of the algorithm fundamental steps is the differentiation step (Step 3 shown in Figure 8, and discussed in Section IV.A.2). Next we discuss whether replacing the basic differentiation operator with the SG filter in the differentiation step improves resulting performance results.

C. SG-BASED DIFFERENTIATION RESULTS

This section investigates the impact the SG filter has when applied as a differentiation operator in step 3 of the pre-processing phase (Step 3 shown in Figure 8, and discussed in Section IV.A.3).

1. SG Filter Differentiation Simulation Settings

Table 4 shows parameter values selected during these tests. Table 1 shows the threshold values taken from the histogram plots in Chapter V, Section B used at each noise level for the primary testing. Threshold values were 0.42 for SNR levels of 12 and 9 dB, 0.2 for 6 and 4 dB, 0.1 for 2 dB, and 0.01 for 0 and -2 dB, respectively. Furthermore items that were discussed in the previous section did not affect the SG filter differentiation results and are not discussed further.

Threshold	**Analysis window Overlap amount (%)**	**Phase Shift time Accuracy Tolerance (% of window length)**	**Frame Size (n)**
See Table 2.	0, 60	5, 10, 15	300

Table 4. SG filter differentiation trial run parameter settings segmentation method B.

2. Difference Filter Differentiation Versus SG Filter Results

Figures 65 and 66 show correct phase shift detection rates obtained for no overlap and 60% window overlapping configurations when the SG filter is used to implement the differentiation step. Comparing these results against those obtained when the basic difference operator is used to implement that step (Figures 52 and 63) shows that applying the SG filter leads to better detection rates between -2 and 6dB SNR at which point both implementations return a 100% correct shift detection rate. This improvement is due to the adaptive noise cancelation aspect of the SG differentiation filter. At lower noise levels (i.e., 4 dB and above) the filter does not have much noise to remove, and the improvement is difficult to notice.

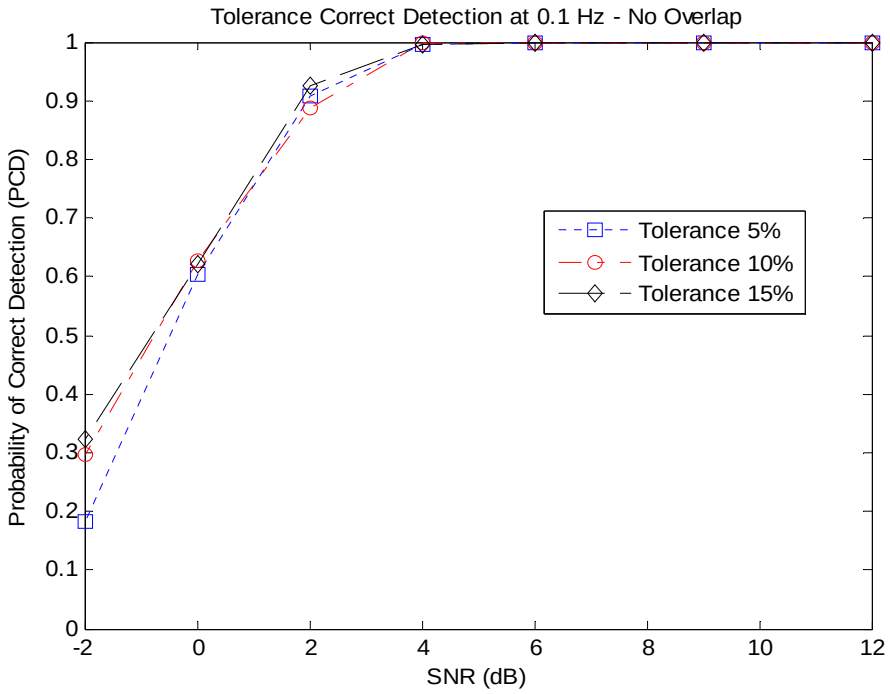


Figure 65. Probability of correct detection obtained for the SG filter differentiation operator, signal normalized carrier frequency = 0.1 Hz, no window overlap, 5% 10% 15% accuracy detection tolerance levels.

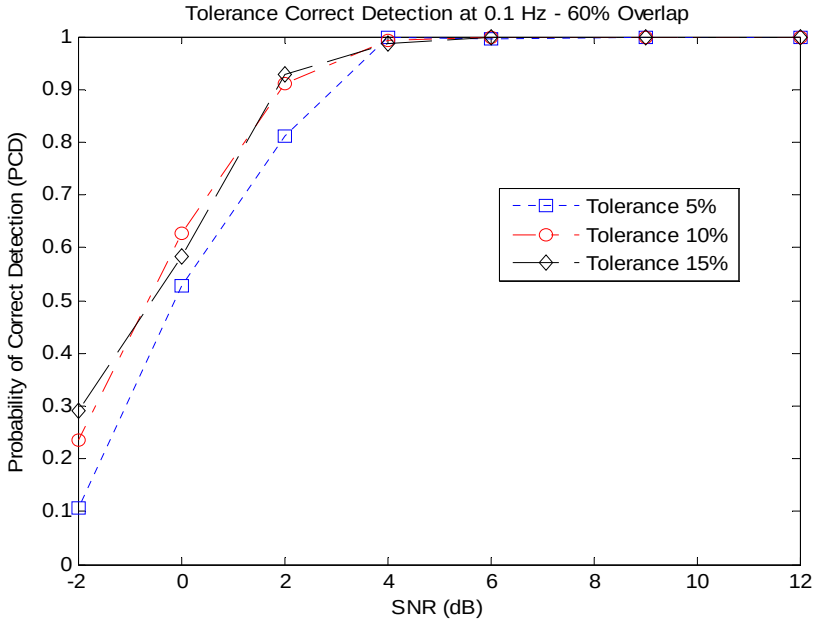


Figure 66. Probability of correct detection obtained for the SG filter differentiation operator, signal normalized carrier frequency = 0.1 Hz, 60% window overlap, 5% 10% 15% accuracy detection tolerance levels.

Figure 67 shows that false alarm probabilities significantly degrade for SNR levels equal to 4dB or lower. Results also show that SG filter implementation leads to higher false alarm rates than the difference filter method at higher noise levels (by comparing Figures 67 and 68 to Figures 54 and 64). These plots show that the SG filter differentiation method reached a false alarm rate equal to 0.9969 at 0 Db, whereas the difference filter method reached 0.9488 false alarm rate at 0 dB, for 10% tolerance, and no-overlapping windows configurations. These results indicate that the implementation is limited to SNR levels above 4 dB SNR. Two reasons exist for the SG filter implementation poor performance. First, recall that the algorithm was specifically designed for the difference filter differentiation step; target images and structuring element definitions were not redefined when the SG filter implementation was investigated. Second, the SG filter was shown to remove phase information as the SNR level decreases, resulting in performance degradations with this implementation in low SNR levels. However, better performance results may be obtained if user-specified parameters were redefined for the SG filter implementation. This investigation is left for future work.

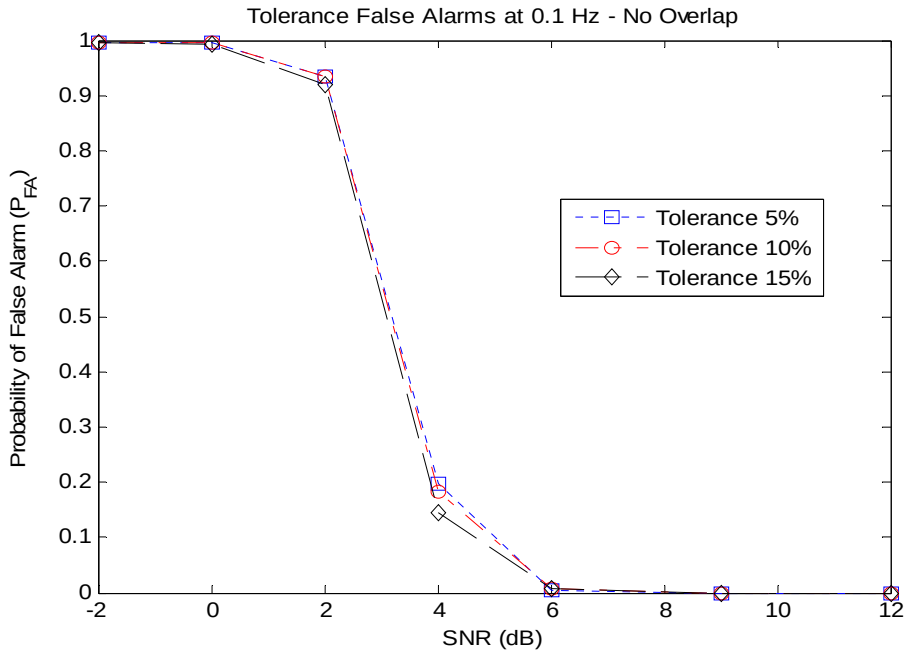


Figure 67. Probability of false alarm obtained for the SG filter differentiation operator, signal normalized carrier frequency = 0.1 Hz, no window overlap, 5% 10% 15% accuracy detection tolerance levels.

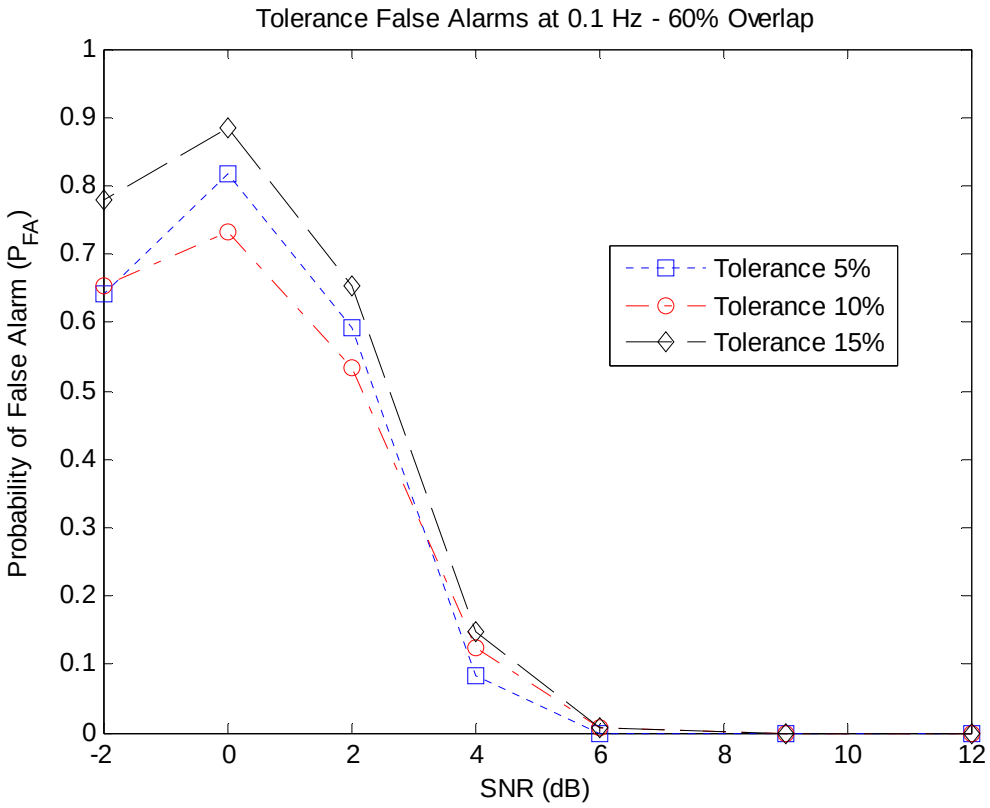


Figure 68. Probability of false alarm obtained for the SG filter differentiation operator, signal normalized carrier frequency = 0.1 Hz, 60% window overlap, 5% 10% 15% accuracy detection tolerance levels.

Figures 69 and 70 present accuracy results obtained for no overlap and window overlapping configurations when using the SG filter implementation of the differentiation operator. Figures 71 and 72 present error rates obtained for the same combinations. Comparing results obtained for the SG filter and the difference filter implementation of the differentiation step show that there are no overall significant improvements by one differentiation implementation over the other.

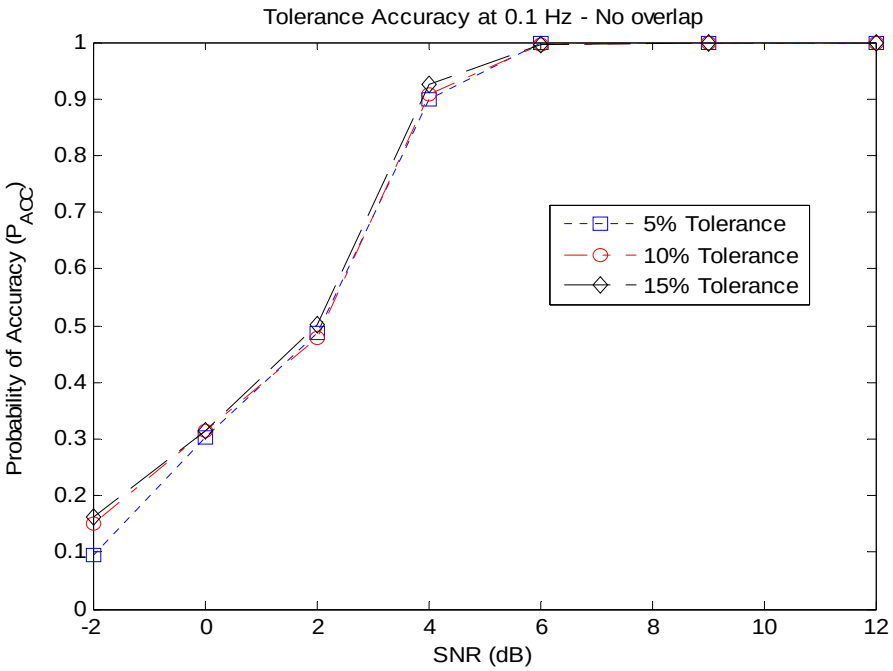


Figure 69. Probability of accuracy obtained for the SG filter differentiation operator, signal normalized carrier frequency = 0.1 Hz, no window overlap, 5% 10% 15% accuracy detection tolerance levels.

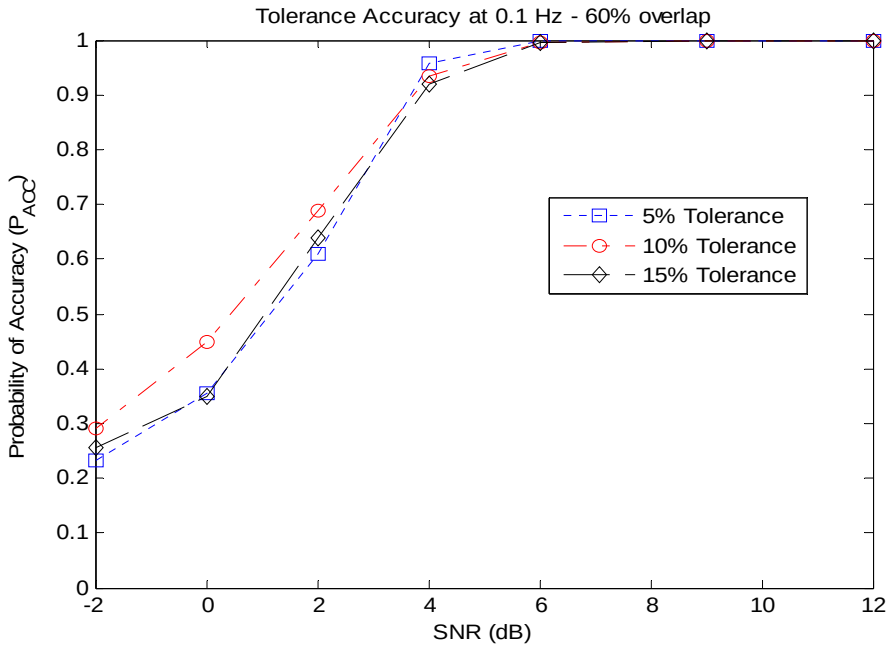


Figure 70. Probability of accuracy obtained for the SG filter differentiation operator, signal normalized carrier frequency = 0.1 Hz, 60% window overlap, 5% 10% 15% accuracy detection tolerance levels.

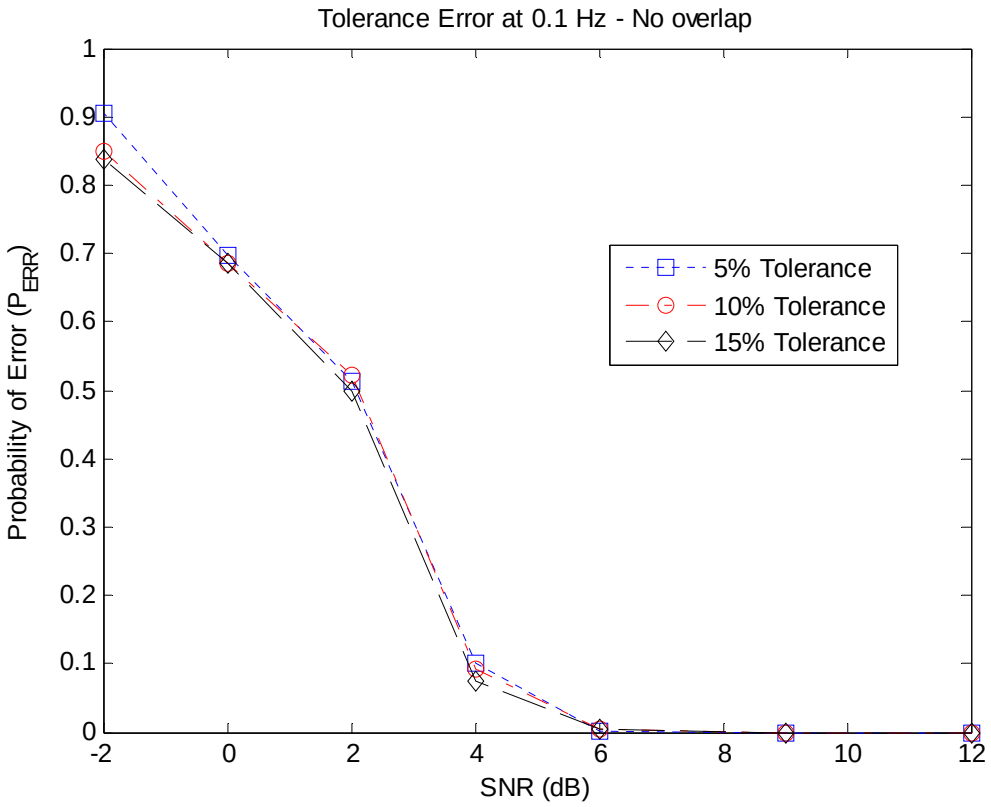


Figure 71. Probability of error obtained for the SG filter differentiation operator, signal normalized carrier frequency = 0.1 Hz, no window overlap, 5% 10% 15% accuracy detection tolerance levels.

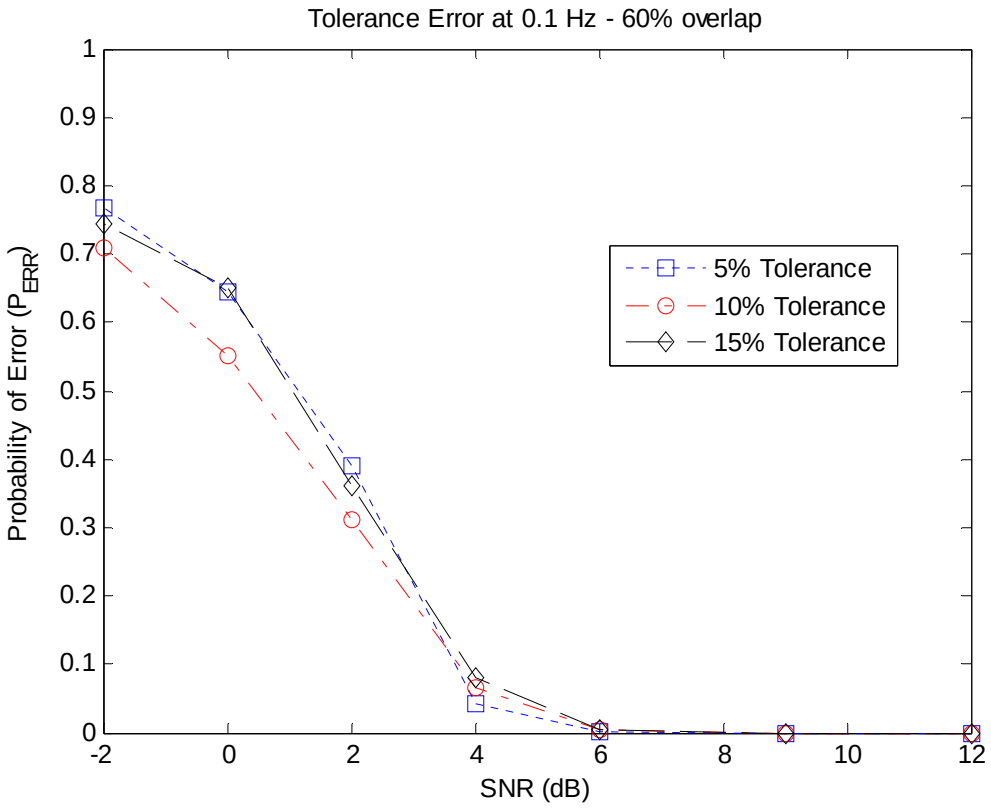


Figure 72. Probability of error obtained for the SG filter differentiation operator, signal normalized carrier frequency = 0.1 Hz, 60% window overlap, 5% 10% 15% accuracy detection tolerance levels.

This section presented detection performance results obtained for the two segmentation and differentiation implementations schemes considered in the study. We also presented results obtained using three different window overlapping ranges, accuracy levels, and for low and high normalized signal carrier frequencies. Overall results show that:

- A data segmentation which leaves no masked data segment leads to better performance results, as to be expected,
- Overlapping windows allow multiple looks of the same data segments, which lead to increase correct detection rates in medium to high overlap ranges,
- Similar results were obtained for the two window lengths investigated,

- The normalized signal carrier frequency had no significant impact on performance results, as a result of the differentiation step (step 3 of the algorithm),
- Results obtained for 5%, 10% and 15% phase shift estimate results were similar for the range of SNR considered,
- The SG implementation of the differentiation filter did not clearly improve performance results over those obtained with the difference filter differentiation step, as user-specified parameters such as filter lengths, structuring masks and threshold values were not optimized for that operator.

D. OPERATING CHARACTERISTIC CURVES

Operating characteristic curves were developed to investigate the sensitivity of the algorithm to threshold changes at specific noise levels. Such curves are useful when interested in selecting "best" thresholds for a desired probability of false alarm and correct detection. Figures 73 to 75 show the operating characteristic for SNR levels of 4, 2 and 0 dB SNR, 10% detection accuracy tolerance, and a no overlapping window configuration using the SG filter implementation of the differentiation step. Plots were generated by increasing the phase shift decision level from 0.05 through 0.95 in 0.05 increments. Figure 73 shows that the optimum threshold for the 4 dB SNR case is in the range between 0.5 and 0.4. In addition, Figure 73 shows that both false alarm and correct detection rates decrease as the threshold value increases. Results also show that the false alarm rate increase and the decision threshold sensitivity increases as the SNR level decreases.

Finally, the trend as noise level increases is shown by all three plots. The operator curve moves from the top-left corner in the 4 dB SNR plot towards the lower-right corner in the 2 dB SNR plot. The sensitivity to threshold values causes the operator curves to spread out as SNR decreases when comparing all figures. Additional plots and results are presented in Appendix B.

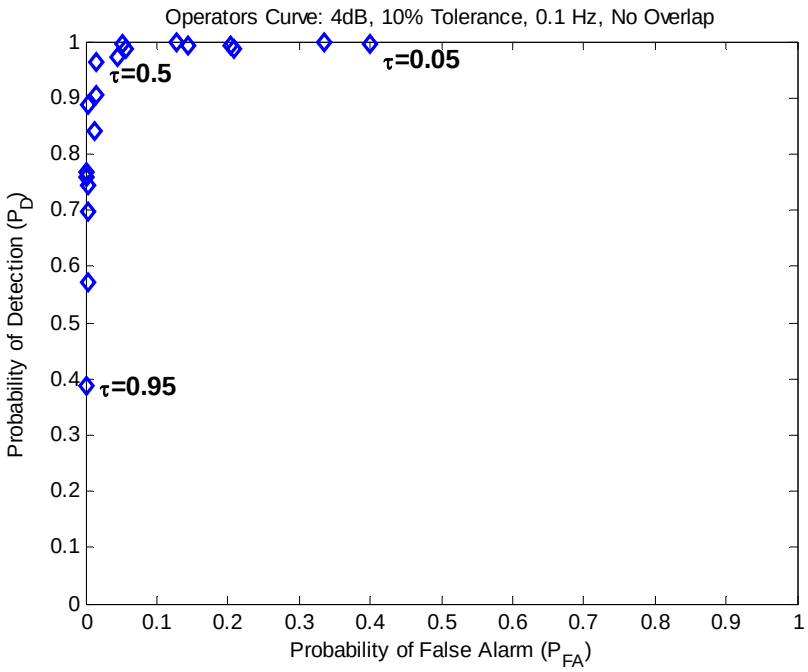


Figure 73. Operating Characteristic Curve, QPSK + AWGN, SNR=4dB, SG filter implementation.

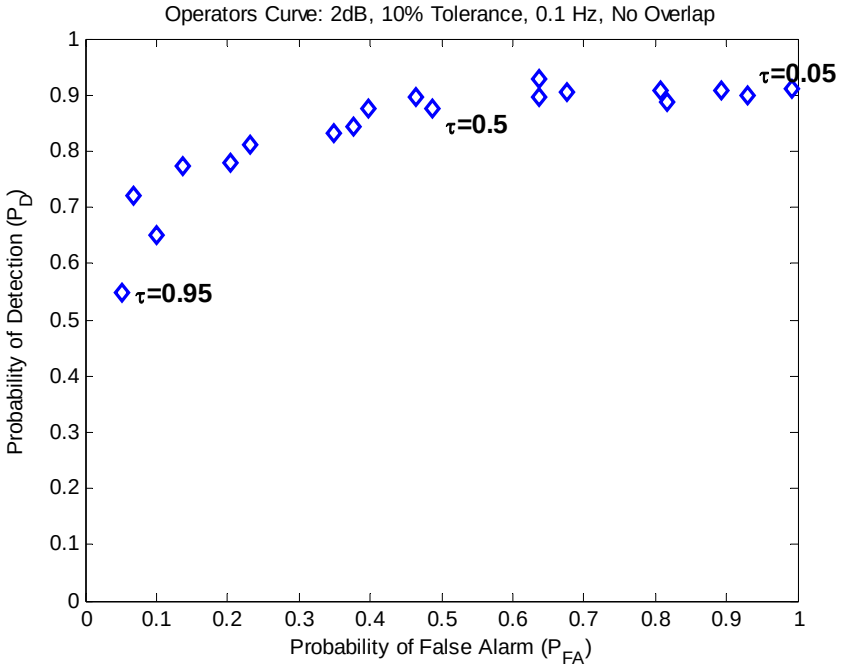


Figure 74. Operating Characteristic Curve, QPSK + AWGN, SNR=2dB, SG filter implementation.

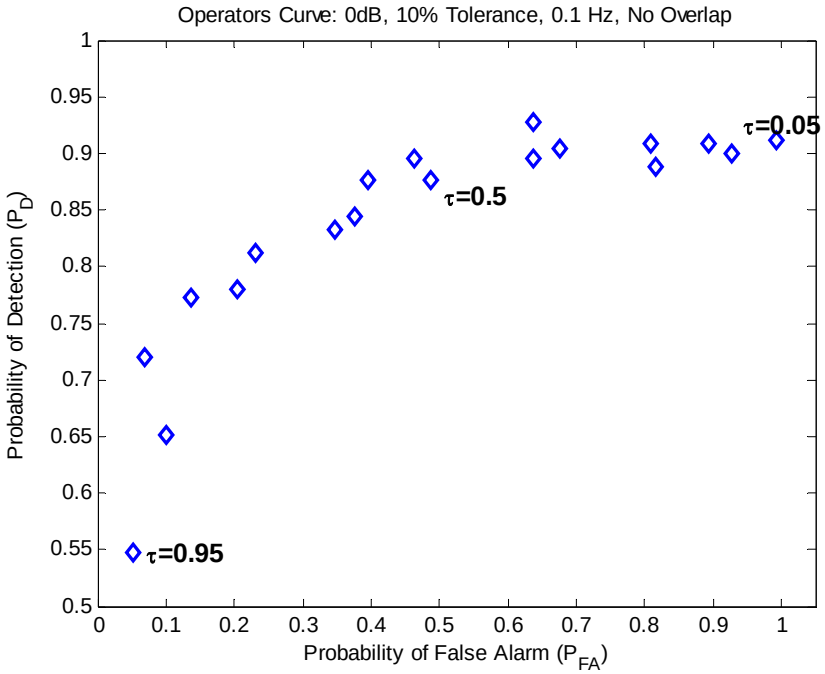


Figure 75. Operating Characteristic Curve, QPSK + AWGN, SNR=0dB, SG filter implementation.

VII. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

This thesis investigated the detection of phase shifts contained in Phase Shift Keyed (PSK) signals using textural image processing and a two-dimensional matched filter approach. The algorithm designed in this work used the information provided by the signal Temporal Correlation Function (TCF) phase to detect the potential presence and location of phase shift times within each frame. One-dimensional filters, differentiation techniques, two-dimensional filters and edge detection schemes performed preliminary image cleaning and isolated the phase shift features of interest. Morphological operators were applied to enhance the phase shift location and a two-dimensional matched filter applied to isolate its location. Results show the proposed scheme performs well down to SNR levels equal to 4dB, with detection and false alarm rates around 1.0 and 0.0841, respectively. Results also show performances degrade significantly below SNR levels less than and equal to 2 dB. Results showed the overall scheme may be quite sensitive to specific selection of parameters and operators used in the pre-processing stage.

The main results are summarized below.

- Results showed that the detection algorithm implemented using the difference filter in the differentiation step has a 100% detection accuracy rate for 10% detection accuracy tolerance level for 60% and 90% window overlapping configurations at 6dB SNR and above. Results also show that a 99.6% detection accuracy rate is obtained when the SG filter is used for the differentiation step and all other parameters stay the same.
- The Segmentation B method resulted in improved performances, as expected as it insured no signal segment was masked during processing. In fact, a 7.6% detection rate increase was noted for SNR levels equal to 4 dB, when using a

60% window overlapping configuration, and 10% detection accuracy tolerance. This improvement is to be expected as it insures there are no signal blind spots.

- Normalized signal carrier frequency values of 0.1 Hz and 0.4 Hz were tested and results showed no noticeable differences, as detection rates were within a few percentage points of each other throughout the range of SNRs tested.
- Shorter window lengths had lower probability of detect and lower false alarm rates for the segmentation option A with no noticeable affect on segmentation option B.
- Adding a window overlapping capability to the decision step proved to be beneficial as it reduced the false alarm rates. This overlapping windows option prevented missed detections for phase shifts located close to the analysis windows ends. Results showed a 96.6% detection rate at 4 dB SNR, for 90% window overlapping configuration, and 10% tolerance, exceeding the identical no-overlap case results equaling 0.954 accuracy rate.
- Results showed that the algorithm yielded higher detection and false alarm rates, as tolerance level was increased from 5% to 15%.

Overall, the phase shift detection and tracking algorithm proved successful and adaptable to various circumstances. The algorithm gives latitude within its programming framework for the user to adjust target images and structuring elements and other factors to enable higher detection rates at higher noise levels as desired for specific circumstances.

B. RECOMMENDATIONS AND FUTURE STUDY

Recommended improvements include the following:

- Mask both ends of the signal image analysis windows prior to the edge detection enhancement phase in order to minimize false alarms from spurious noise bits remaining following the two-dimensional mean filter noise removal step.
- Mask the top 32x30 pixels of the final signal image vice cropping to permit further scanning by the 2-D cross-correlation process further increasing detection rates.
- Note that target images and morphological structuring elements were optimized for 6 dB SNR noise level while using the difference derivative approximation. Significant improvement may be realized by optimizing the program at a higher noise level (e.g. 4 dB) and using the SG filter in the differentiation stage of the algorithm because of its noise removal capabilities.
- Investigate adapting this algorithm for use with FSK vice PSK schemes and investigate resulting detection performances.
- Investigate minimum bit encoding period restrictions on this algorithm.
- And, ultimately, test the algorithm in a hardware implementation where actual analog PSK or FSK signals are analyzed.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A – MATLAB SOURCE CODE

The MATLAB code for the phase shift detection algorithm is listed below in hierarchical order including the final SIMULINK model. The model and code presented use segmentation option B and the difference filter algorithm settings, however, the SG filter function *diffsg.m* is listed below to permit its use in place of the *diff.m* MATLAB function for differentiation step. Functions referenced in the code but not shown below are basic MATLAB internal functions or contained in the *Communications*, *Signal Processing*, or *Image Processing Toolboxes* [9].

A. MAIN SIMULATION PROGRAM – WINDOW MATCH

```
%% MAIN PROGRAM - WINDOW MATCH
% PURPOSE: PSK DETECTION SCHEME USING 2D CROSS-CORRELATION MATCHED FILTER
%   This simulation program was designed to allow the user to set up the
%   number of runs desired as well as specify parameters.
%   Details of the input parameters are listed below next to the variables themselves.
%   INPUTS(Defined in Variable initialization section):
%       SNR - Desired signal-to-noise-ratio for simulated QPSK signal in dB.
%       THRESHOLD - 2D cross-correlation threshold value (Rx>Thresh => Shift, Rx<Thresh => No
Shift).
%       POVLP - Amount of window that overlaps previous window in % (0 or >=50% possible, common
values are 0,50%,60%,80%,90%).
%       SLEN - Desired length of simulated QPSK signal.
%       NOSH - Desired number of phase shifts for this simulation.
%       WLEN_U - User desired window length.
%       ERTOL - Error tolerance in percent (values tested were 5,10,& 15%).
%       FSIG - Normalized frequency for simulated QPSK signal in Hz (between 0 and 0.5 Hz).
%   OUTPUTS:
%       POUT - vector of final program shift times (Program Output)
%       POUTF - vector of false alarm times (Program False Output)
%       AGREED - array, column one is binary yes or no if POUT time agrees
%       with AST by +/-COMP, column two shows the AST value for those POUTs
%       that agree with AST.
%       AGREEDF - array, column one is binary yes or no if POUTF time does
%       not agree with AST by +/-COMP, column two shows the corresponding
%       POUTF value.
%       ERRSTATS - Listing of run statistics..(see ERRSTAT.M).
%
%   USES: PATMATCH.MDL, SIGNALGEN, TCF, DTRUTH, ERRSTAT, DIFFSG (AS
%   APPLICABLE), MEANFILT, MATCHFILES.MAT.
%
%   Filename: windowmatch.m
%
%                   LT Joseph Kramer, REVISION DATE: 25AUG2009
```

```
 format compact
%% Variable Initialization
snrt = [12 9 6 4 2 0 -2]';
tau=[.28 .28 .28 .45 .5 .8 .98];
load tablet     % Loads a list file for long testruns called 'table'.
load matchfiles % Stores program target images and structuring elements.
for item=1:1    % Outer loop used to run tests from a list called 'table', loaded from 'tablet'.
   ck = [];     % Track noshiftck runs
   ph = [];     % Track mxval through runs
   out = [];    % Track output data
   stout=[];    % Track error statistic data
   finstat=[];  % Track complete run data
    for run=1:length(snrt)       % Inner loop where SNR is varied for sets of other test parameters in 'table'.
       snr = snrt(run,1);       % Test Signal SNR in dB (VARIABLE)
       temp = find(snrt==snr);   % Look up appropriate threshold per SNR when not called from 'table'.
       sensval = tau(temp);      % No Shift Image Decision Threshold (VARIABLE)
       thresh = sensval;
       povlp = 0;               % Percent of overlap: [overlap = povlp*wlen] (VARIABLE)
       slen = 6000;              % Simulated Signal length in discrete time units (VARIABLE)
       nosh = 5;                % Simulated Signal Number of Phase Shifts spread throughout signal length
(VARIABLE)
       wlen_u = 300;             % Program window length (VARIABLE) set by user.
       edg_offset=64;            % Window edge offset length.
       indexoff=60;             % Timing index for SIMULINK model - Change when changing segmentation
modes. (30 SEG A, 60 SEG B)
       wlen=wlen_u+edg_offset;   % Program frame size accounting for any edge offset.
       offset2 = round((wlen-wlen_u)/2)-1;  % Endcap offset minus one for indexing purposes.
       nwolp = round(1/(1-povlp));         % Number of overlapping windows that can see a given shift time.
       ertol = .1;              % Error Tolerance - used to compute COMP based on window length
(VARIABLE)
       fsig = 0.1;              % Simulated Signal Normalized Frequency (Hz)(VARIABLE)
       comp = wlen_u*ertol;      % number of time units on either side of actual that are acceptable.
%% PSK Test Signal Generation.
[inputsig ast phdat] = signalgen(snr,fsig,slen,nosh,povlp,wlen,offset2+1);     %Simulated PSK Signal
generation
%% Parsing Segment: Parse input signal taking into account the overlap and segmentation method.
seg = [];         % Segmentation Vector Initialization
% Find window starting point step value based on overlap and desired window length (wlen_u).
if nwolp==1
   owvlp = 0;
   st=wlen_u;   % Window Starting Point Step Value
else
ovlp = round(wlen_u/nwolp); %150 pixels for window length 300 at 50% Overlap.
st = ovlp;       % Window Starting Point Step Value
end

% Parsing and Truth vector Loop
truth=[];         % Truth vector based on overlap.
stad=[];          % Start addressing vector for use in time indexing later.
nolp=0;           % Count Number of Overlap Periods
for k = 1:st:slen
   if (k+wlen-1)>slen  % Stop if end of window exceeds signal length.
      break
   else
   seg = cat(2,seg,inputsig(k:k+wlen-1));  % Pull segment from signal and store in 'seg'.
```

```
   ext = 0;
   for r=1:length(ast)     % Establish truth vector based on overlaps.
   if (k+offset2-1<=ast(r))&&(ast(r)<=k+wlen-offset2)
      truth = cat(1,truth,1);
      ext = 1;
      break
   end
   end
   if ext==0              % If a shift does not exist in a time frame store a 0 in 'truth' vector.
     truth = cat(1,truth,0);
   end
   stad = cat(1,stad,k);   % Store the starting point of each window segment used during analysis.
   nolp=nolp+1;
   end
end

%% Image Processing and 2-D Cross-Correlation Portion
% Cropping masks for Signal Image
if wlen-wlen_u==0
   coordim = [1, 1, 29, wlen-1];                % Segmentation A is used - crop to width of 30 pixels but do not
crop the ends off the image.
else
   coordim = [1, offset2+1, 29, wlen-offset2+1];  % Segmentation B is used - crop to width of 30 pixels and
crop the leading end off the image.
end
   maskends = zeros(32,30);                     % Zero mask to remove trailing boundary condition from signal
images.
%% Target Images and Image Processing Structuring Elements.
  % Define target images used for 2-D cross-correlation.
   tgt = setri4b;              % Target 1
   ima = sum(sum(tgt));       % IMA used for Croxx-correlation Normalization Factor in SIMULINK
   tgt2 = setri4d;             % Target 2
   imb = sum(sum(tgt2));       % IMA used for Croxx-correlation Normalization Factor in SIMULINK
  % Define various Dilation and Erosion templates used in processing.
   sz = 24;
   sz2 = 12;
   sed = strel(e2);
   se1 = strel(diag(ones(sz,1)));
   se2 = strel(fliplr(diag(ones(sz,1))));
   se = strel(ones(8));
   se3 = strel(w);
%% Find Window Shift Times by Call to SIMULINK model
  % Track various SIMULINK model outputs defined below.
   shft=[]; mxval=[]; mxtgta=[]; mxtgtb=[];
   imsto1 = []; imstomn = []; c1=[]; p1=[]; point=[];
for m=1:wlen:length(seg)            % Scan each window segment
   [im pyo] = tcf(seg(m:m+wlen-1)); % Computes TCF per window segment.
  % Signal image noise removal and phase shift transformation portion.
   im = imcrop(im,coordim);        % Crop signal image to Mx30 dimensions (M = WLEN for SEG A, or
=WLEN-32 for SEG B).
   im = medfilt1(im,15);           % One-dimensional MEDIAN filter along time axis (N=15th Order).
   im=diff(im,2,1);                % Differentiation step using DIFF.M for difference filter algorithm or
diffsg(im,8,9) for SG filter;
   im=round(im);
   im=meanfilt(im,3);              % Two-dimensional MEAN filter dimensions (3x3).
```

```
   [im] = edge(im,'roberts',[0.1],'nothinning');   % Roberts edge detection operation
  % Cascade dilation/erosion set 1: First, noise removal and phase shift transformation step.
   im = imdilate(im,sed);      % Dilation operation.
   im1 = imerode(im,se1);      % Erosion operations.
   im2 = imerode(im,se2);
   im=sign(im1+im2);           % Superposition of both individual erosions.
  % Cascade dilation/erosion set 2: Second, detailed noise removal step.
   im = imdilate(im,se);       % Dilation operation.
   im = imerode(im,se3);       % Erosion operation.
  % Masking operation.
   im(length(im)-32+1:length(im),:)=maskends;
   sim('patmatch');            % Call to SIMULINK model.
   shft=cat(1,shft,simout);    % Relative window phase shift time from SIMULINK model
   point=cat(1,point,pointer+1);  % Flag the image that gave max correlation from SIMULINK model
   c1=cat(4,c1,corn1);         % Raw cross-correlation values prior to normalization from SIMULINK model
   p1=cat(4,p1,corn);          % Raw cross-correlation values prior to normalization from SIMULINK model
   mxval=cat(1,mxval,mxvalout); % Final MAX cross-correlation value from SIMULINK model
   mxtgta=cat(1,mxtgta,mxtgt);  % Target 1's MAX cross-correlation value from SIMULINK model
   mxtgtb=cat(1,mxtgtb,mxtgt2); % Target 2's MAX cross-correlation value from SIMULINK model
end
%% Apply No-Shift threshold following call to SIMULINK model.
zer = shft>0;               % Threshold shifts are assigned 1, no shifts are assigned 0.
final = (shft+stad).*zer;    % Program shift times after adjusting for window positions.
[pout poutf agreed agreedf]=dtruth(final,truth,stad,wlen_u,nwolp,ast,comp);  % Performs final program
decision and overlapping rules as applicable.
%% ERROR STATS PORTION OF PROGRAM - THIS IS FOLLOWING ALL PROGRAM OUTPUT.
% This portion is only for testing and analysis purposes.
%% Trial Info Summary: Tolerance and Data Dump for various parameters.
sprintf('Run tolerance: %0.2g%%',ertol*100)                  % Display the tolerance level of the current run.
[errstats] = errstat(ast,agreed(:,1),agreedf(:,1),slen,wlen_u);  % Calculates accuracy and error statistics for
current run.
disp('   %H   %Pfa   %M   %Pdet  %Pacc  %Perr')               % Display these accuracy and error statistics
for current run.
disp('                                        ')
disp(errstats)
%% Data Collection section - Save applicable data run header and information....
% Saves name and appends SNR level to each file name - ensure SNR changes
% or change naming convention so you don't overwrite.
nm = ['filename_goes_here',num2str(snr)];
rundat = [snr, thresh, nosh, slen/wlen, round(nwolp), errstats];
finstat = [finstat; rundat];
   end
   save(nm,'finstat');
end
```

B. CALL SIMULATION GENERATION PROGRAM

```
%% FUNCTION [Y S PHDAT]=SIGNALGEN(SNR,F,TOTLEN,NOS,POVLP,WLEN,OFST)
%% PURPOSE: GENERATES A SIGNAL OF SIGNIFICANT LENGTH
%% WITH VARIOUS PHASE SHIFT OCCURRENCES FOR TESTING
%   INPUTS:
%       SNR - vector of possible shift times from windowmatch.m
%       F - binary vector representing no shifts based on correlation
```

```
%      threshold
%      TOTLEN - scalar number showing number of windows that overlap given shift time.
%      NOS - vector of actual shift times generated during windowmatch
%   OUTPUTS:
%      Y - vector of final program shift times (Program Output)
%      S - array, column one is binary yes or no if POUT time agrees
%
%   Filename: signalgen.m
%                  LT Joseph Kramer, Revision Date: 20AUG2009
function [y s phdat]=signalgen(snr,f,totlen,nos,povlp,wlen,ofst)
[y s phdat] = PSKgen(f,totlen,nos,povlp,wlen,ofst);
y=awgn(y,snr);
```

C. PSK GENERATION FUNCTION

```
%% FUNCTION [X ST PHDAT]=PSKGEN(F,N,NUMSH,POVLP,WLEN,OFST)
% Signal Generator for Frequency Shift Signal
% [x]=PSKgen(n,nosh,t)
% Generate a Binary Phase Shift Keying signal length n with normalized f [0 to .5].
% nosh is number of shifts within the signal...
% t sets the time index the phase shift occurs.
% ps = phase shift.
%                                                    LT Joseph Kramer 22JAN2009
function [x st phdat]=PSKgen(f,n,numsh,povlp,wlen,ofst)
nosh=numsh;
A = sqrt(2);  % Input signal is normalized to 1 watt/0dbw.
if nosh==0    % If no shifts inserted.
   ti = [0:1:n-1];
   x = A*cos(2*pi*f*ti);
   st=0;
   phdat=0;
   return
elseif nosh<10
   base = 380;
else
   base = round(wlen);       % Range of available shift spacings.
end

for k = 1:nosh               % Generates semi-random shift times
   num = rand(1);
   frac = num-mod(num,0.1);
   if k==1
   st(k,:) = round(base*.89);
   else
   st(k,:) = round(st(k-1)+2*(base-2*ofst));
   end
end

if povlp==0               % Code added to ensure no overlap runs allow correct results.
   if n/nosh<wlen         % If using zero overlap make sure the shifts will fit 1 to a window.
      disp('Invalid number of shifts for signal length and window length.')
      disp('Number of shifts adjusted to signal length divided by window length.')
      clear st
      nosh=floor(n/wlen);
```

```
    end
   for k=1:nosh          % With no overlap can have only 1 shift per window and no edge shifts.
      if k==1
         st(k) = round(wlen*.5);
      else
         st(k) = st(k-1)+2*(wlen-2*ofst);    % Adjust spacing with offset in mind.
      end
   end
end
Ts=1;
x =[];
phdat=phrand();
for g = 2:nosh+1       % Generate QPSK shift magnitudes.
    int = phrand();
    while int==phdat(g-1)   % Ensure there is a shift when inserted.
       int = phrand();
    end
    phdat(g,1)=int;
end
% Use phase shifts and shift times to generate a QPSK signal.
for m=1:length(st)
    if m==1     % First time vector (odd + phase)...
       ti = [m-1:Ts:st(m)];
       ps = phdat(m);
       a = A*cos(2*pi*f*ti+ps);
       x = cat(2,x,a);
    elseif rem(m,2)==1      % ODD +phase intermediate time vectors...
       ti = [st(m-1)+1:Ts:st(m)];
       ps = phdat(m);
       a = A*cos(2*pi*f*ti+ps);
       x = cat(2,x,a);
    elseif rem(m,2)==0      % EVEN (no phase) int. vectors...
       ti = [st(m-1)+1:Ts:st(m)];
       ps = phdat(m);
       a = A*cos(2*pi*f*ti+ps);
       x = cat(2,x,a);
    end
end
% Complete vector to length n using appropriate phase.
if rem(m+1,2)==1    % Last time vector...
   ti = [st(m)+1:Ts:n-1];
   ps = phdat(m+1);
   a = A*cos(2*pi*f*ti+ps);
   x = cat(2,x,a);
elseif rem(m+1,2)==0
   ti = [st(m)+1:Ts:n-1];
   ps = phdat(m+1);
   a = A*cos(2*pi*f*ti+ps);
   x = cat(2,x,a);
end
```

D. TEMPORAL CORRELATION FUNCTION

```
%% FUNCTION [Y PY]=TCF(X);
%% PURPOSE: PRODUCE TCF OUTPUT MATRIX (IMAGE) GIVEN PORTION OF SIGNAL.
%   INPUTS:
%       X - Signal in vector format Nx1 dimension.
%   OUTPUTS:
%       Y - TCF Output image dimensions NxN/2.  Output will have time axis
%       on Y-axis of length N and Lags along X-axis of length N/2.
%       PY - Raw TCF output prior to phase computation.
%   Filename: tcf.m
%                   LT Joseph Kramer, Revision Date: 1MAY2009
function [y py]=tcf(x)
sc=hilbert(x);                          % complex shift signal
Rx=zeros(length(sc),round(length(sc)/2)-1);     % preallocation Rx
for tau=0:1:(length(sc)/2-1)   % tau index
   for t=1:length(sc)        % time index
      if (t-tau) >0 && (t+tau)<= length(sc)   % (t-tau/2) >0 since causal
                                    % (t+tau/2)<= length(sc)
                                    % index cannot be over
                                    % data range
         Rx(t,tau+1)=sc(t+tau)*conj(sc(t-tau));
      else
         Rx(t,tau+1)=0;
      end
   end
end
py = Rx;
y=angle(Rx);
```

E. DIFFERENTIATION USING SAVITZKY-GOLAY FIR FILTER

```
%% FUNCTION [Y1 B G]=DIFFSG(IMIN,N,F)
%% PURPOSE: Differential Smoothing Function using SGOLAY MATLAB function...
%   This function uses sgolay.m to generate the polynomial filter size N
%   of frame size F and applies the selected derivative to the image IMIN.
%  INPUTS:
%         IMIN - Image file of dimensions (RxC)
%         N - Polynomial order for filter strength. (SCALAR).
%         F - Frame size for filter. (SCALAR: NOTE F>N)
%  OUTPUTS:
%         Y1 - Output image. (RxC)
%         B - Filter weights (NxN).
%         G - Filter coeficients (NxN).
%
%   Filename: diffsg.m
%                   LT Joseph Kramer, REVISION DATE: 7AUG2009
function [y1 b g]=diffsg(imin,N,F)
[b,g] = sgolay(N,F);   % Calculate S-G coefficients
y0=[]; y1=[]; y2=[];
[r c]=size(imin);
HalfWin  = ((F+1)/2) -1;
for j=1:c
```

```
   y = imin(:,j);
   ti=length(y);
for n = (F+1)/2:ti-(F+1)/2
% 1st differential
  SG1(n) =   dot(g(:,2), y(n - HalfWin: n + HalfWin));
end
   y1 = [y1 SG1'];
end
```

F. 2-D MEAN FILTER

```
%% FUNCTION [IM]=MEANFILT(Y,S)
%% PURPOSE: 2-D REGION MEAN FILTER FOR NOISE REMOVAL
%   INPUTS:
%       Y - input image size (RxC)
%       S - odd size of filter (ie.. 3x3 or 5x5)
%   OUTPUT:
%       IM - filtered image buffered to RxC size with row/col of zeros
function [imo]=meanfilt(y,s)
m = (s-1)/2;    % Solve for index variable M
[r c]=size(y);  % Find dimensions of the image
box = ones(s)/(s^2);  % Use a square operator to perform the 2D mean operation.
for k=m+1:r-m
   for j=m+1:c-m
      imo(k,j)=sum(sum(y(k-m:k+m,j-m:j+m).*box));
   end
end
[r2 c2]=size(imo);
if r2~=r
   imo(r2+(r-r2),:)=zeros(1,c2);  % Pad last row with zeros for edge discontinuities.
end
if c2~=c
   imo(:,c2+(c-c2))=zeros(r,1);   % Pad last column with zeros for edge discontinuities.
end
```

G. SIMULINK MODEL

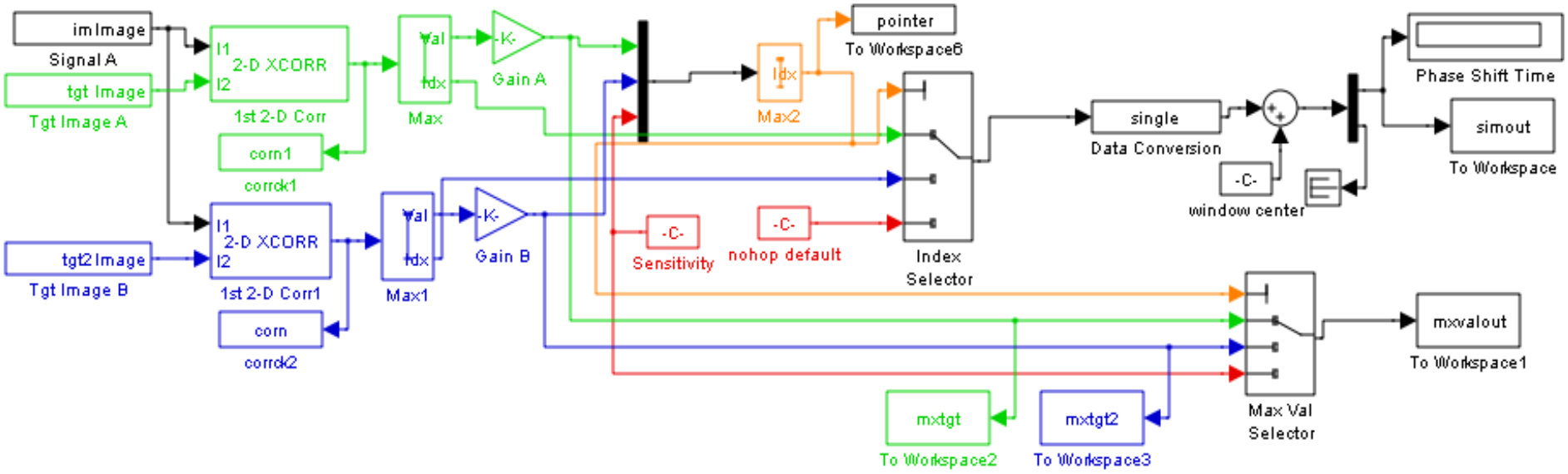


Figure 76. Full Two-dimensional Matched Filter Simulink Model Block Diagram.

H. DECISION MODULE AND OVERLAP IMPLEMENTATION

```
%% FUNCTION [POUT POUTF1 AGREED AGREEDF]=
%    DTRUTH(FINAL,TRUTH,STAD,WLEN,NWOLP,AST,COMP)
%% PURPOSE: OVERLAPPING WINDOW SEARCH & VERIFICATION OF SHIFT TIMES
%   This function uses overlapping windows as primary decision rule
%   regarding perceived program shift times.
%   INPUTS:
%      FINAL - vector of possible shift times from windowmatch.m
%      TRUTH - binary vector developed in Windowmatch.m to show where shifts were inserted.
%      NWOLP - scalar number showing number of windows that overlap given shift time.
%      AST - vector of actual shift times generated during windowmatch call to signalgen.m
%      COMP - scalar value equal to tolerance*window length.
%   OUTPUTS:
%      POUT - vector of final program shift times (Program Output)
%      POUTF1 - vector of false alarm times (~Program Output)
%      AGREED - array, column one is binary yes or no if POUT time agrees
%      with AST by +/-COMP, column two shows the AST value for those POUTs
%      that agree with AST.
%      AGREEDF - array, column one is binary yes or no if POUTF time does
%      not agree with AST by +/-COMP, column two shows the corresponding POUTF value.
%
%   Filename: dtruth.m
%                  LT Joseph Kramer, REVISION DATE: 25AUG2009
function [pout poutf1 agreed agreedf]=dtruth(final,truth,stad,wlen,nwolp,ast,comp)
%% Set step size and zero selectivity based on the NWOLP...
if nwolp==2
   nwolp=2;
   d=1;              % zero selectivity setting - if >=d zeros are in a comparison frame - that frame is not
ignored...
elseif nwolp==3
   d = 2;            % zero selectivity setting - if >=d zeros are in a comparison frame - that frame is not
ignored...
   f = 2;            % Choose between 2,3 or 4 out of a group for the decision...
elseif nwolp<10
   nwolp=round(nwolp);
   d = 3;            % zero selectivity setting - if >=d zeros are in a comparison frame - that frame is not
ignored...
   f = 3;            % Choose between 2,3 or 4 out of a group for the decision...
elseif nwolp==10
   d = 5;            % zero selectivity setting - if >=d zeros are in a comparison frame - that frame is not
ignored...
   f = 4;            % Choose between 2,3 or 4 out of a group for the decision...
elseif nwolp <=20
   d = 10;           % zero selectivity setting - if >=d zeros are in a comparison frame - that frame is not
ignored...
   f = 4;            % Choose between 2,3 or 4 out of a group for the decision...
end
ptime = final;       % Set window times that do not meet threshold check to zero.
pout = [0];          % Program output variable - ie.. the program estimated shift times.
agreed = [];         % Stats verification array - not a real program output - for simulation results calculation
only.
% Loop based on overlap to calculate the program decisions.
if nwolp~=1          % Special case for no - overlap - just select the window based on threshold...
```

```
 if nwolp==2          % Special case for only 50% overlap...only compare 1st 2 and pick 1/2...
    for m=1:1:length(ptime)-nwolp
       x = ptime(m:m+nwolp-1); % Choose Window group based on step size (NWOLP)
       rdif = [];
       if sum(x==0)>=d
          pout=cat(1,pout,0);
          continue
       elseif x(1)==0
          pout=cat(1,pout,0);
          continue
       else
          for k = 1:nwolp          % Loop calculates the relative difference within
             dif = abs(x-x(k));    % each window group for minimum difference decision
             rdif = cat(2,rdif,dif);
          end
          sdif = sum(rdif);        % Gives a vector sum showing the item with minimum difference to have
smallest values
          [dfv dfp]=sort(sdif);    % Sorts this vector (sdif) so that the minimum 2 or 3 values can be
selected based on comparison.
          if (abs(x(dfp(1))-x(dfp(2)))<=comp)   % Accomplishes best 2 or 3 out of NWOLP comparison
using COMP.
             bstval=x(dfp(1));
             if (bstval>=stad(m))&&((stad(m)+wlen-1)>bstval)
                if dfp(1)~=1
                   pout=cat(1,pout,0);
                else
                pout=cat(1,pout,bstval);
                end
             else
                pout=cat(1,pout,0);
             end
          else
             pout=cat(1,pout,0);
          end
       end
    end
 else
    for m=1:1:length(ptime)-nwolp
       x = ptime(m:m+nwolp-1); % Choose Window group based on step size (NWOLP)
       rdif = [];
       if sum(x==0)>=d
          pout=cat(1,pout,0);
          continue
       elseif x(1)==0
          pout=cat(1,pout,0);
          continue
       else
          for k = 1:nwolp          % Loop calculates the relative difference within
             dif = abs(x-x(k));    % each window group for minimum difference decision
             rdif = cat(2,rdif,dif);
          end
          sdif = sum(rdif);        % Gives a vector sum showing the item with minimum difference to have
smallest values
          [dfv dfp]=sort(sdif);    % Sorts this vector (sdif) so that the minimum 2 or 3 values can be
selected based on comparison.
```

```
            if (abs(x(dfp(1))-x(dfp(2)))<=comp)&&(abs(x(dfp(1))-x(dfp(f-1)))<=comp)&&(abs(x(dfp(1))-
x(dfp(f)))<=comp)   % Accomplishes best 2 or 3 out of NWOLP comparison using COMP.
                bstval=x(dfp(1));
                y = x==bstval;
                if (bstval>=stad(m))&&((stad(m)+wlen-1)>bstval)
                   if (stad(m+dfp(1)-1)-1<x(dfp(1)))&&(x(dfp(1))<=stad(m+nwolp-1)+wlen-1)
                      pout=cat(1,pout,bstval);
                   else
                      pout=cat(1,pout,0);
                   end
                else
                   pout=cat(1,pout,0);
                end
             else
                pout=cat(1,pout,0);
             end
          end
       end
    end
  pout = pout(2:length(pout));
else
   pout = ptime.*(ptime>0);
end

   %% Filter duplicate times out of pout and maintain lineup with truth vector.....
   pout2=[];
   for l=1:length(pout)
      if pout(l)==0
         pout2 = [pout2; 0];
         continue
      elseif length(find(abs(pout(l)-pout2)<=comp))>0   % Don't repeat shift time answers within window
tolerance (comp).
         pout2 = [pout2; 0];
         continue
      else
         pout2 = [pout2; pout(l)];                  % store accompanying program shift time
      end
   end
   pout=pout2(1:length(pout2));
   if length(pout)<length(truth)
      pout = [pout; zeros(length(truth)-length(pout),1)];
   elseif length(truth)<length(pout)
      truth = [truth; zeros(length(pout)-length(truth),1)];
   end
   poutf = pout.*~truth;
%% Filter Duplicate Pout times based on COMP...
   poutf2=[0];
   for l=1:length(poutf)
      if poutf(l)==0
         continue
      elseif length(find(abs(poutf(l)-poutf2)<=comp))>0    % Don't repeat shift time answers within window
tolerance (comp).
         continue
      else
```

```
        poutf2 = [poutf2; poutf(l)];                % sets logic bit to 1 and stores accompanying actual shift
time
      end
   end
   if (length(poutf2)==1)&&(poutf2==0)
      poutf1 = poutf2;
   else
      poutf1=poutf2(2:length(poutf2));
   end
%% STATS CHECK ONLY - NOT VALID PROGRAM OUTPUT
% Agreed shows 1 for program times matching actual within COMP limit.(HIT)
%            0 for program times that don't match (FA or MISS)

if isempty(pout)
   pout =0;
   agreed = [0 0 0];
else
   agreed = [0 0 0];
   for l=1:length(pout)
      for k = 1:length(ast)
         if ((abs(pout(l)-ast(k)))<=comp)   % Loop finds phase shift times that are within COMP,
           % for accuracy analysis...
            if length(find(abs(pout(l)-agreed(:,2))<=comp))>0  % Don't repeat shift time answers within
window tolerance (comp).
              continue
            else
              agreed =[agreed;[1 ast(k) pout(l)]];          % sets logic bit to 1 and stores accompanying actual
shift time
            end
          end
      end
   end
end
[r c]=size(agreed);
agreed = agreed(2:r,:);

%% Sort poutf to find and count distinct false alarm times that do not appear in AST.
% AgreedF shows 1 for program times that don't match (FA or MISS)
%            0 for program times matching actual within COMP limit.(HIT)
    agreedf = [0 0];
    if isempty(poutf1(poutf1>0))
       poutf = 0;
    else
    poutf = poutf1(poutf1>0);
    end
    poutf = [poutf; poutf(length(poutf))+5; 0;];
   for l=1:length(poutf-1)
      for k = 1:length(ast)
        if poutf(l)==0
           continue
        elseif (((abs(poutf(l)-ast(k)))>comp)&(poutf(l)<poutf(l+1))) % Loop finds phase shift times that are
within COMP,
            if length(find(abs(poutf(l)-agreedf)<=comp))>0         % Don't repeat shift time answers within
window tolerance (comp).
              continue
```

```
            else
              agreedf = [agreedf; [1 poutf(l)]];              % sets logic bit to 1 and stores accompanying
actual shift time
            end
          end
        end
    end
    [r2 c]=size(agreedf);
    agreedf = agreedf(2:r2,:);
end
```

I. STATISTICS CALCULATIONS

```
%% FUNCTION [ESTATS]=ERRSTAT(A,CP,FP,SL,WL)
%% PURPOSE: CALCULATE THE CORRECT HITS, FALSE ALARMS, AND MISSES FOR A RUN.
%   Uses known shift vector to calculate the accuracy rates of the algorithm.
%   INPUTS:
%          A - Truth vector based on test signal shift times. (1's & 0's)
%          [AST]
%          CP - Correct Program shift times that agree with T. (1's & 0's) [AGREED(:,1)]
%          FP - False Program shift times (1's & 0's) [AGREEDF(:,1)]
%          Detects.
%          SL - Signal Length
%          WL - Window Length
%   OUTPUTS:
%          ESTATS - Vector that tracks the stats for this run. (in percent)
%          [PCD(1|1) PFA(1|0) PM(0|1) PD(1|1) PACC PERR]
%
%   Filename: errstat.m
%
%                    LT Joseph Kramer, REVISION DATE: 28JUL2009
function [estats]=errstat(a,cp,fp,sl,wl)
format short g;
a = a>0;                    % Forms a binary vector based on program shift times.
cp = cp>0;
fp = fp>0;
if isempty(cp)
    cp=0;
end
if isempty(fp)
    fp=0;
end
nonh = (sl-64)/wl-sum(a);       % Possible no shifts in this run.
TP = sum(cp);                % Correct detection of Shifts P(1|1).
if length(fp)>nonh
    FN = sum(a)-TP;            % MISS...
    FP = nonh;                % All NONHs are False Alarms in this case...
    TN = 0;                   % If all NONHs are FAs then there are no (0|0)s
else
    FP = sum(fp);              % Type I error. (False Alarm) P(1|0)
    FN = sum(a)-TP;            % Type II error. (Miss) P(0|1)
    if nonh==0
       TN = 0;
```

```
   else
      TN = nonh-sum(fp);      % For accuracy P(0|0)
   end
end
PD = TP;
PA = (TP+TN)/(sum(a)+nonh);     % Paccuracy = {P(1|1)+P(0|0)}/TOTAL
PE = (FP+FN)/(nonh+sum(a));     % Perror = {P(0|1)+P(1|0)}/TOTAL
%%            Pcd(1|1) Pfa(1|0) Pm(0|1)   Pd(1|1) Pacc  Perr
estats = 100*[TP/sum(a) FP/nonh FN/sum(a) PD/sum(a) PA PE];
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B – DATA AND PLOTS

A. THRESHOLD PLOTS

1. DIFFERENCE FILTER ALGORITHM

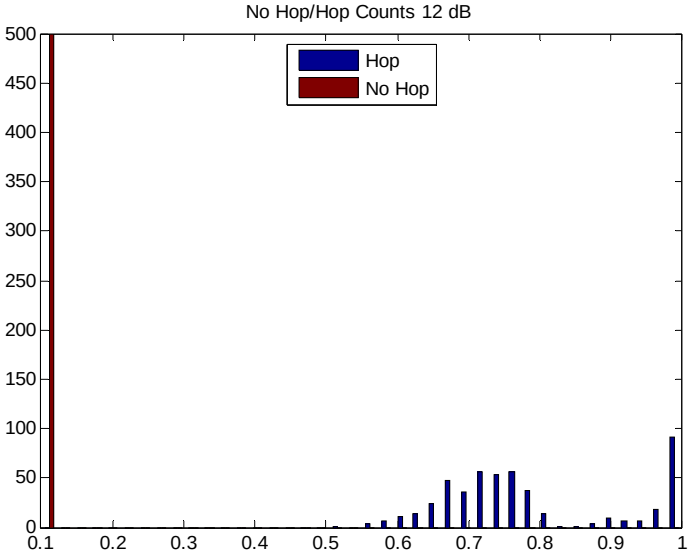


Figure 77. Cross-correlation values for QPSK + AWGN, SNR = 12 dB, using Difference filter, phase shift and no-shift cases.

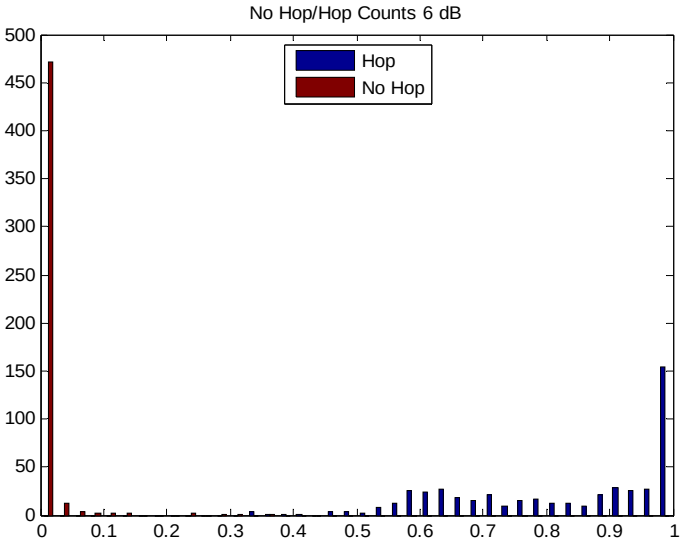


Figure 78. Cross-correlation values for QPSK + AWGN, SNR = 6 dB, using Difference filter, phase shift and no-shift cases.

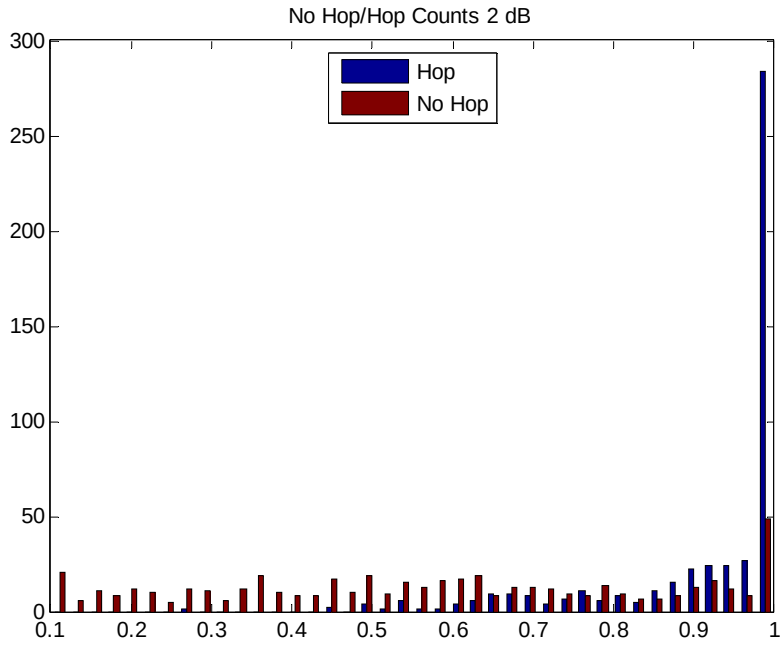


Figure 79. Cross-correlation values for QPSK + AWGN, SNR = 2 dB, using Difference filter, phase shift and no-shift cases.

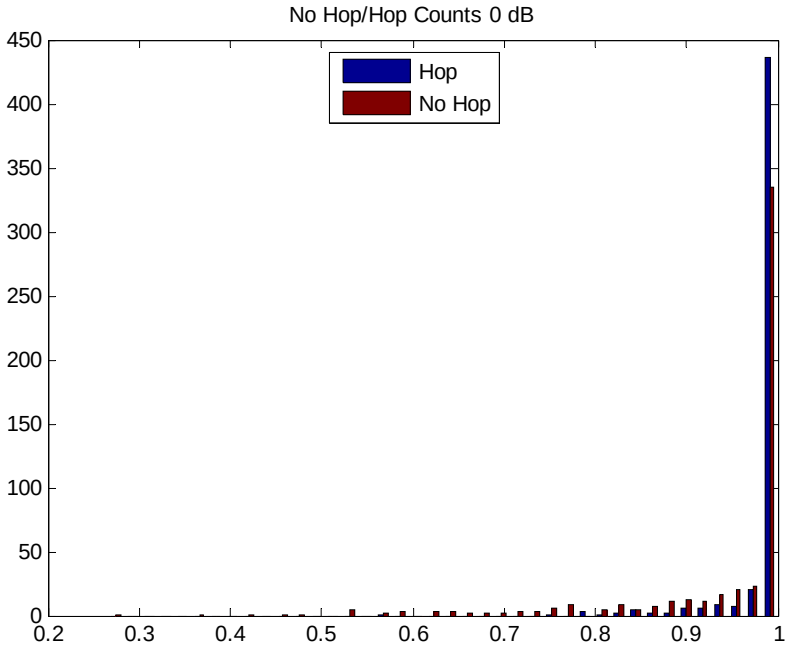


Figure 80. Cross-correlation values for QPSK + AWGN, SNR = 0 dB, using Difference filter, phase shift and no-shift cases.

2. SG FILTER

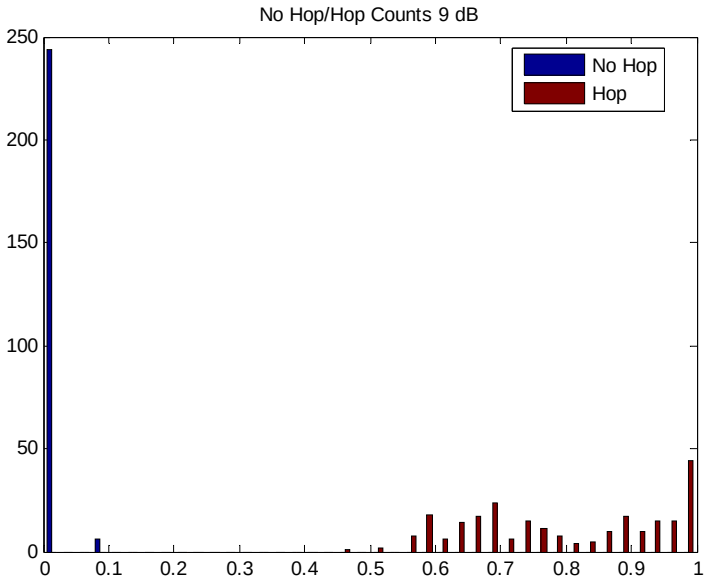


Figure 81. Cross-correlation values for QPSK + AWGN, SNR = 9 dB, using SG filter, phase shift and no-shift cases.

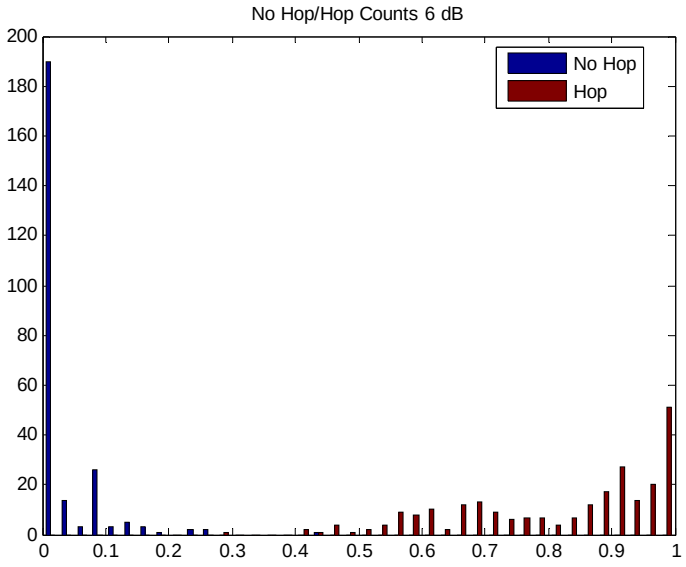


Figure 82. Cross-correlation values for QPSK + AWGN, SNR = 6 dB, using SG filter, phase shift and no-shift cases.

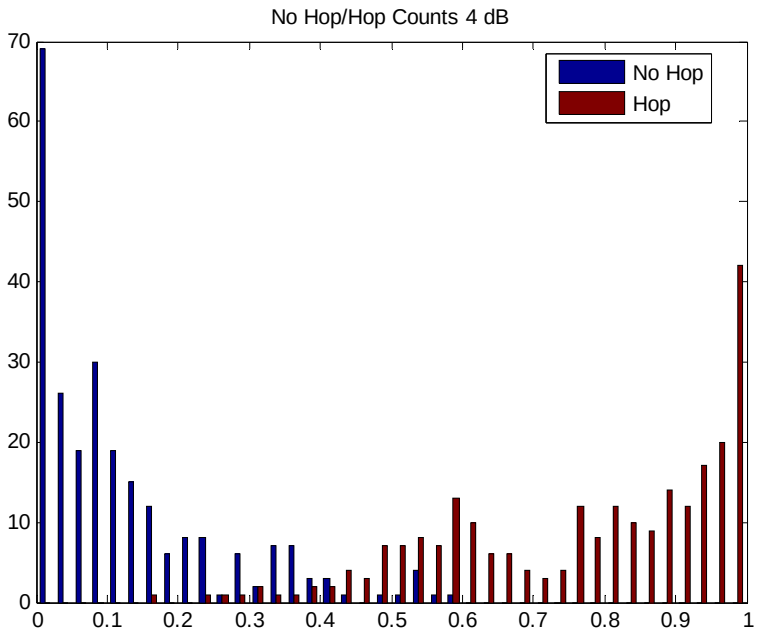


Figure 83. Cross-correlation values for QPSK + AWGN, SNR = 4 dB, using SG filter, phase shift and no-shift cases.

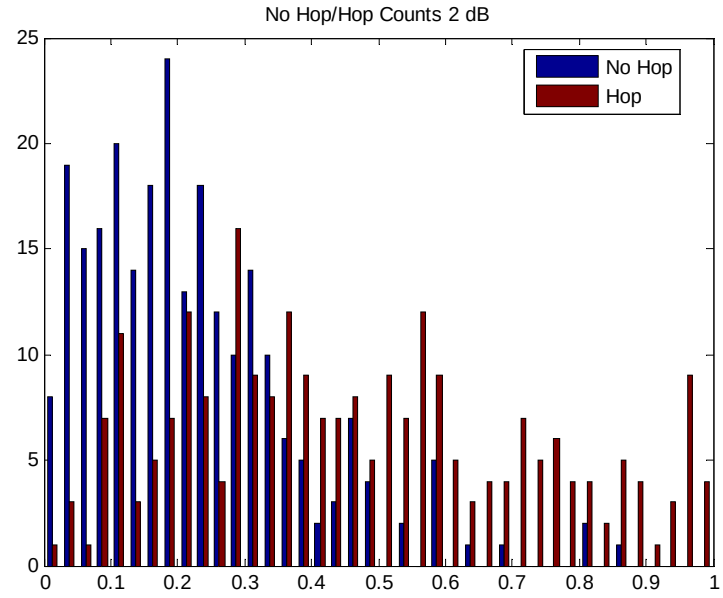


Figure 84. Cross-correlation values for QPSK + AWGN, SNR = 2 dB, using SG filter, phase shift and no-shift cases.

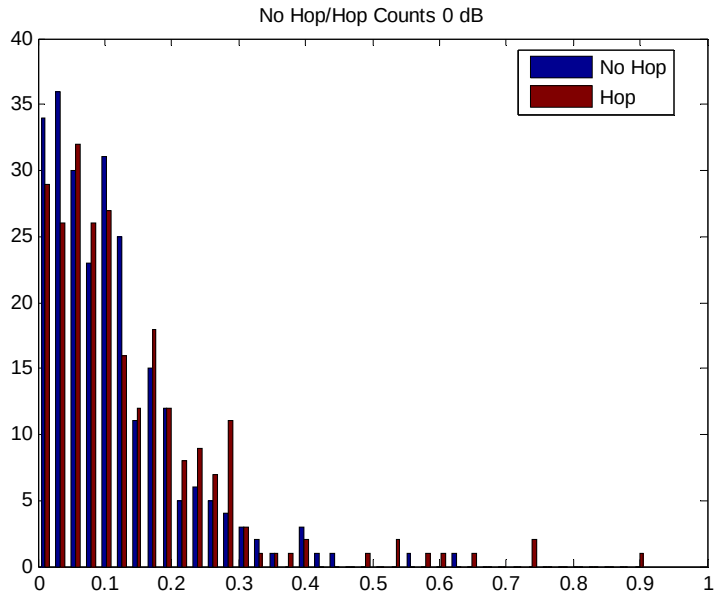


Figure 85. Cross-correlation values for QPSK + AWGN, SNR = 0 dB, using SG filter, phase shift and no-shift cases.

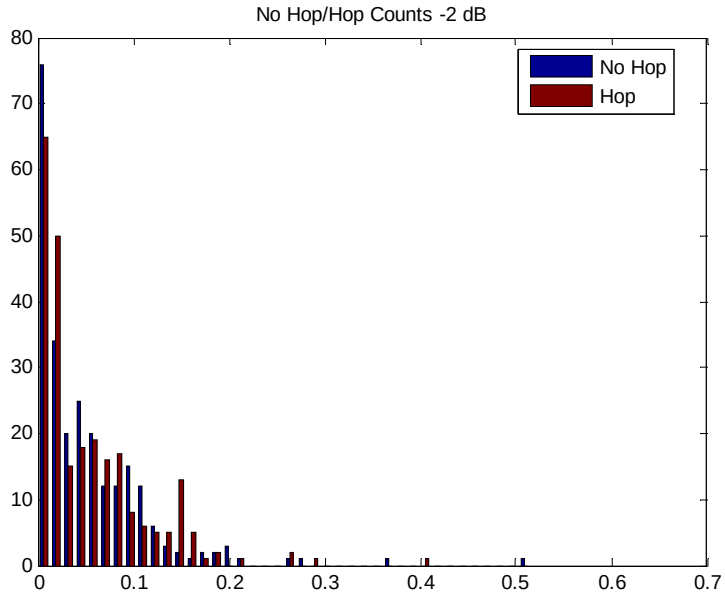


Figure 86. Cross-correlation values for QPSK + AWGN, SNR = -2 dB, using SG filter, phase shift and no-shift cases.

B. SIMULATION DATA

Note, data displayed is at a frequency of 0.1Hz, window length 300, tolerance of 10% unless otherwise stated in the tables.

Segmentation A - Difference Filter				Segmentation B - Difference Filter			
	Overlap Rate				Overlap Rate		
SNR (dB)	0%	60%	90%	SNR (dB)	0%	60%	90%
PCD							
12	1.0000	1.0000	1.0000	12	1.0000	1.0000	1.0000
9	1.0000	1.0000	1.0000	9	1.0000	1.0000	1.0000
6	0.9960	0.9920	0.9960	6	1.0000	1.0000	1.0000
4	0.9760	0.9000	0.9480	4	0.9760	0.9760	0.9720
2	0.8800	0.7840	0.8320	2	0.8280	0.7480	0.8320
0	0.5640	0.3640	0.5880	0	0.4800	0.3760	0.4840
-2	0.2120	0.1200	0.2480	-2	0.1160	0.1000	0.2000
Probability of False Alarm							
12	0.0000	0.0000	0.0000	12	0.0000	0.0000	0.0000
9	0.0000	0.0000	0.0000	9	0.0000	0.0000	0.0000
6	0.0040	0.0080	0.0040	6	0.0160	0.0000	0.0000
4	0.0560	0.0560	0.0641	4	0.0681	0.0961	0.0400
2	0.6245	0.5685	0.6486	2	0.7246	0.5204	0.5244
0	0.8968	0.7206	1.0000	0	0.9488	0.7647	1.0000
-2	0.9848	0.3643	0.8848	-2	0.9928	0.3843	0.8848
Probability of Accuracy							
12	1.0000	1.0000	1.0000	12	1.0000	1.0000	1.0000
9	1.0000	1.0000	1.0000	9	1.0000	1.0000	1.0000
6	0.9960	0.9920	0.9960	6	0.9920	1.0000	1.0000
4	0.9600	0.9220	0.9420	4	0.9540	0.9400	0.9660
2	0.6278	0.6078	0.5918	2	0.5518	0.6138	0.6539
0	0.3337	0.3217	0.2941	0	0.2657	0.3057	0.2421
-2	0.1136	0.3777	0.1817	-2	0.0616	0.3577	0.1576
Probability of Error							
12	0.0000	0.0000	0.0000	12	0.0000	0.0000	0.0000
9	0.0000	0.0000	0.0000	9	0.0000	0.0000	0.0000
6	0.0040	0.0080	0.0040	6	0.0080	0.0000	0.0000
4	0.0400	0.0780	0.0580	4	0.0460	0.0600	0.0340
2	0.3722	0.3922	0.4082	2	0.4482	0.3862	0.3461
0	0.6663	0.6783	0.7059	0	0.7343	0.6943	0.7579
-2	0.8864	0.6223	0.8183	-2	0.9384	0.6423	0.8424

Table 5. Difference filter differentiation results for 0%, 60%, and 90% overlap rates and segmentation options A and B.

Window = 256 - Difference Filter			Freq. = 0.4Hz - Difference Filter		
	Overlap Rate			Overlap Rate	
SNR (dB)	0%	60%	SNR (dB)	0%	60%
PCD					
12	1.0000	1.0000	12	0.9960	1.0000
9	1.0000	1.0000	9	1.0000	1.0000
6	1.0000	1.0000	6	1.0000	1.0000
4	0.9480	0.9720	4	0.9880	0.9560
2	0.8280	0.8240	2	0.8280	0.7840
0	0.4600	0.3880	0	0.4520	0.3560
-2	0.1640	0.1160	-2	0.1440	0.0840
Probability of False Alarm					
12	0.0000	0.0000	12	0.0000	0.0000
9	0.0000	0.0000	9	0.0000	0.0000
6	0.0000	0.0030	6	0.0080	0.0040
4	0.0447	0.0596	4	0.0560	0.0480
2	0.7120	0.5064	2	0.7366	0.5565
0	0.9324	0.7507	0	0.9608	0.7366
-2	0.9890	0.4528	-2	0.9969	0.3803
Probability of Accuracy					
12	1.0000	1.0000	12	0.9980	1.0000
9	1.0000	1.0000	9	1.0000	1.0000
6	1.0000	0.9983	6	0.9960	0.9980
4	0.9522	0.9539	4	0.9660	0.9540
2	0.5185	0.6346	2	0.5458	0.6138
0	0.2351	0.3085	0	0.2457	0.3097
-2	0.0763	0.3631	-2	0.0736	0.3517
Probability of Error					
12	0.0000	0.0000	12	0.0020	0.0000
9	0.0000	0.0000	9	0.0000	0.0000
6	0.0000	0.0017	6	0.0040	0.0020
4	0.0478	0.0461	4	0.0340	0.0460
2	0.4815	0.3654	2	0.4542	0.3862
0	0.7649	0.6915	0	0.7543	0.6903
-2	0.9237	0.6369	-2	0.9264	0.6483

Table 6. Difference filter differentiation frequency results for 0% and 60% overlap rates and window length equal to 256.

Difference Approximation Differentiation - Segmentation B								
	12 dB	9 dB	6 dB	4 dB	2 dB	0 dB	2 dB	Tolerance
PCD								
0% Overlap	1.0000	1.0000	1.0000	0.9640	0.7960	0.3440	0.0880	5%
	1.0000	1.0000	1.0000	0.9760	0.8280	0.4800	0.1160	10%
	1.0000	1.0000	1.0000	0.9640	0.8240	0.5320	0.1480	15%
60% Overlap	1.0000	1.0000	1.0000	0.9200	0.7280	0.2600	0.0560	5%
	1.0000	1.0000	1.0000	0.9760	0.7480	0.3760	0.1000	10%
	0.9960	1.0000	1.0000	0.9720	0.8080	0.4360	0.1640	15%
Probability of False Alarm								
0% Overlap	0.0000	0.0000	0.0040	0.0761	0.6886	0.9168	0.9928	5%
	0.0000	0.0000	0.0160	0.0681	0.7246	0.9488	0.9928	10%
	0.0000	0.0000	0.0040	0.0641	0.6926	0.9328	0.9969	15%
60% Overlap	0.0000	0.0000	0.0040	0.0560	0.4804	0.7206	0.3163	5%
	0.0000	0.0000	0.0000	0.0961	0.5204	0.7647	0.3843	10%
	0.0000	0.0000	0.0040	0.0841	0.4884	0.8567	0.4484	15%
Probability of Accuracy								
0% Overlap	1.0000	1.0000	0.9980	0.9440	0.5538	0.2137	0.0476	5%
	1.0000	1.0000	0.9920	0.9540	0.5518	0.2657	0.0616	10%
	1.0000	1.0000	0.9980	0.9500	0.5658	0.2997	0.0756	15%
60% Overlap	1.0000	1.0000	0.9980	0.9320	0.6238	0.2697	0.3697	5%
	1.0000	1.0000	1.0000	0.9400	0.6138	0.3057	0.3577	10%
	0.9980	1.0000	0.9980	0.9440	0.6599	0.2897	0.3577	15%
Probability of Error								
0% Overlap	0.0000	0.0000	0.0020	0.0560	0.4462	0.7863	0.9524	5%
	0.0000	0.0000	0.0080	0.0460	0.4482	0.7343	0.9384	10%
	0.0000	0.0000	0.0020	0.0500	0.4342	0.7003	0.9244	15%
60% Overlap	0.0000	0.0000	0.0020	0.0680	0.3762	0.7303	0.6303	5%
	0.0000	0.0000	0.0000	0.0600	0.3862	0.6943	0.6423	10%
	0.0020	0.0000	0.0020	0.0560	0.3401	0.7103	0.6423	15%

Table 7. Difference filter differentiation results for 0% and 60% overlap rates across 5%,10%, and 15% tolerance.

Savitzky-Golay Differentiation - Segmentation B								
	12 dB	9 dB	6 dB	4 dB	2 dB	0 dB	2 dB	Tolerance
PCD								
0% Overlap	1.0000	1.0000	1.0000	0.9960	0.9080	0.6040	0.1840	5%
	1.0000	1.0000	1.0000	1.0000	0.8880	0.6280	0.2960	10%
	1.0000	1.0000	1.0000	0.9960	0.9240	0.6200	0.3240	15%
60% Overlap	1.0000	1.0000	0.9960	1.0000	0.8120	0.5280	0.1080	5%
	1.0000	1.0000	1.0000	0.9920	0.9120	0.6280	0.2360	10%
	1.0000	1.0000	1.0000	0.9880	0.9280	0.5840	0.2920	15%
Probability of False Alarm								
0% Overlap	0.0000	0.0000	0.0040	0.1962	0.9328	0.9969	0.9969	5%
	0.0000	0.0000	0.0080	0.1842	0.9328	0.9969	0.9969	10%
	0.0000	0.0000	0.0080	0.1441	0.9208	0.9928	0.9969	15%
60% Overlap	0.0000	0.0000	0.0000	0.0841	0.5925	0.8167	0.6405	5%
	0.0000	0.0000	0.0080	0.1241	0.5325	0.7326	0.6526	10%
	0.0000	0.0000	0.0080	0.1481	0.6526	0.8848	0.7807	15%
Probability of Accuracy								
0% Overlap	1.0000	1.0000	0.9980	0.9000	0.4878	0.3037	0.0936	5%
	1.0000	1.0000	0.9960	0.9080	0.4778	0.3157	0.1496	10%
	1.0000	1.0000	0.9960	0.9260	0.5018	0.3137	0.1636	15%
60% Overlap	1.0000	1.0000	0.9980	0.9580	0.6098	0.3557	0.2337	5%
	1.0000	1.0000	0.9960	0.9340	0.6899	0.4478	0.2917	10%
	1.0000	1.0000	0.9960	0.9200	0.6378	0.3497	0.2557	15%
Probability of Error								
0% Overlap	0.0000	0.0000	0.0020	0.1000	0.5122	0.6963	0.9064	5%
	0.0000	0.0000	0.0040	0.0920	0.5222	0.6843	0.8504	10%
	0.0000	0.0000	0.0040	0.0740	0.4982	0.6863	0.8364	15%
60% Overlap	0.0000	0.0000	0.0020	0.0420	0.3902	0.6443	0.7663	5%
	0.0000	0.0000	0.0040	0.0660	0.3101	0.5522	0.7083	10%
	0.0000	0.0000	0.0040	0.0800	0.3622	0.6503	0.7443	15%

Table 8. SG filter differentiation results for 0% and 60% overlap rates across 5%,10%, and 15% tolerance

C. OPERATING CURVES DATA AND PLOTS

1. CURVE DATA

Note, data displayed is for a normalized signal carrier frequency equal to 0.1Hz, window length 300, detection accuracy tolerance equal to 10% unless otherwise stated in the tables.

Difference filter algorithm Operating Characteristic Curve Data								
SNR	**6 dB**		**4 dB**		**2 dB**		**0 dB**	
τ	**PCD**	**False Alarm**	**PCD**	**False Alarm**	**PCD**	**False Alarm**	**PCD**	**False Alarm**
0.05	1.0000	0.0320	0.9840	0.5805	0.8480	0.9969	0.4520	0.9969
0.10	1.0000	0.0160	0.9880	0.4724	0.8160	0.9969	0.4560	0.9969
0.15	1.0000	0.0080	0.9920	0.3163	0.8320	0.9728	0.4360	0.9969
0.20	1.0000	0.0120	0.9800	0.2762	0.8440	0.9368	0.4560	0.9969
0.25	1.0000	0.0120	0.9960	0.2202	0.8360	0.9288	0.4400	0.9969
0.30	0.9920	0.0000	0.9880	0.2362	0.8360	0.8447	0.3880	0.9969
0.35	0.9920	0.0000	0.9840	0.1201	0.7960	0.7967	0.4560	0.9969
0.40	0.9920	0.0000	0.9640	0.1201	0.8400	0.8127	0.4400	0.9969
0.45	0.9880	0.0000	0.9520	0.0841	0.8000	0.7647	0.4760	0.9969
0.50	0.9600	0.0000	0.9560	0.0601	0.8120	0.7286	0.5160	0.9969
0.55	0.9280	0.0000	0.9000	0.0560	0.8360	0.6245	0.4560	0.9969
0.60	0.9240	0.0000	0.9000	0.0280	0.8000	0.5405	0.4680	0.9928
0.65	0.8360	0.0000	0.8400	0.0280	0.8200	0.5685	0.4560	0.9969
0.70	0.7920	0.0000	0.7800	0.0160	0.7680	0.4444	0.5040	0.9568
0.75	0.6680	0.0000	0.7440	0.0080	0.7880	0.4124	0.4160	0.9528
0.80	0.6280	0.0000	0.7360	0.0080	0.6840	0.3043	0.4440	0.9248
0.85	0.4880	0.0000	0.6280	0.0160	0.7000	0.2482	0.4600	0.9248
0.90	0.3880	0.0000	0.5960	0.0120	0.5920	0.2202	0.4520	0.9368
0.95	0.2880	0.0000	0.4080	0.0000	0.5280	0.1401	0.4760	0.7847

Table 9. Difference filter differentiation operating curve results for input signal SNR levels of 6, 4, 2, and 0 dB.

SG filter Operating Characteristic Curve Data								
SNR	**6 dB**		**4 dB**		**2 dB**		**0 dB**	
τ	**PCD**	**False Alarm**	**PCD**	**False Alarm**	**PCD**	**False Alarm**	**PCD**	**False Alarm**
0.05	1.0000	0.0200	0.9960	0.4003	0.9120	0.9928	0.6360	0.9969
0.10	1.0000	0.0120	1.0000	0.3363	0.9000	0.9288	0.6480	0.9969
0.15	1.0000	0.0040	0.9880	0.2082	0.9080	0.8928	0.6200	0.9969
0.20	1.0000	0.0080	0.9920	0.2042	0.9080	0.8087	0.6360	0.9969
0.25	1.0000	0.0000	1.0000	0.1281	0.8880	0.8167	0.6000	0.9969
0.30	1.0000	0.0000	0.9920	0.1441	0.9040	0.6766	0.5840	0.9928
0.35	1.0000	0.0000	0.9880	0.0560	0.8960	0.6365	0.6800	0.9928
0.40	0.9960	0.0000	0.9960	0.0520	0.9280	0.6365	0.5640	0.9888
0.45	0.9920	0.0000	0.9720	0.0440	0.8960	0.4644	0.6560	0.9808
0.50	0.9680	0.0000	0.9640	0.0160	0.8760	0.4884	0.6400	0.9928
0.55	0.9360	0.0000	0.9040	0.0160	0.8760	0.3963	0.5920	0.9608
0.60	0.9000	0.0000	0.8880	0.0040	0.8440	0.3763	0.6040	0.9488
0.65	0.8000	0.0000	0.8400	0.0120	0.8320	0.3483	0.6400	0.9448
0.70	0.7760	0.0000	0.7680	0.0000	0.8120	0.2322	0.6720	0.8767
0.75	0.6680	0.0000	0.7440	0.0040	0.7800	0.2042	0.6320	0.8367
0.80	0.6520	0.0000	0.7600	0.0000	0.7720	0.1361	0.6200	0.7847
0.85	0.5280	0.0000	0.6960	0.0040	0.7200	0.0681	0.6440	0.7326
0.90	0.4480	0.0000	0.5720	0.0040	0.6520	0.1001	0.5840	0.6686
0.95	0.3680	0.0000	0.3880	0.0000	0.5480	0.0520	0.5800	0.5685

Table 10. SG filter differentiation operating curve results for input signal SNR levels of 6, 4, 2, and 0 dB

2. OPERATING CURVES FOR THE DIFFERENCE FILTER IMPLEMENTATION OF THE DIFFERENTIATION STEP

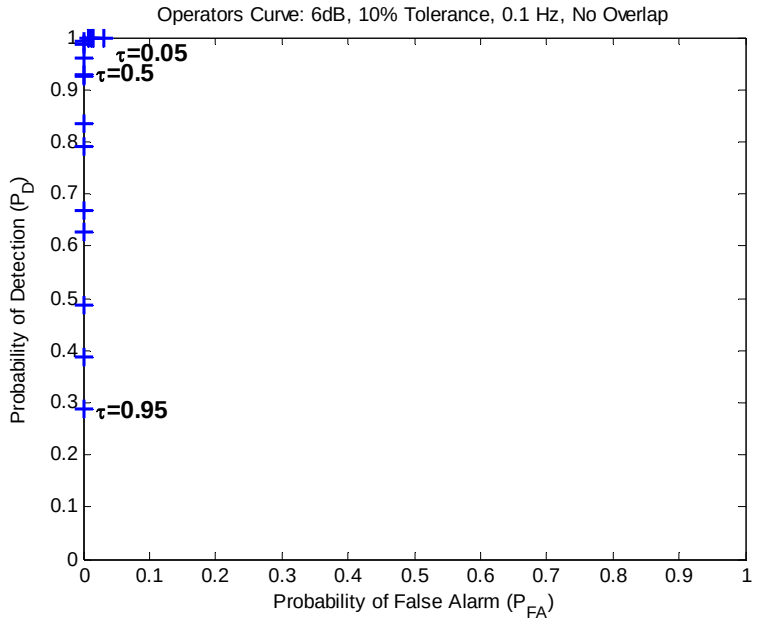


Figure 87. Operating Characteristic Curve, QPSK + AWGN, SNR = 6dB, Difference filter implementation.

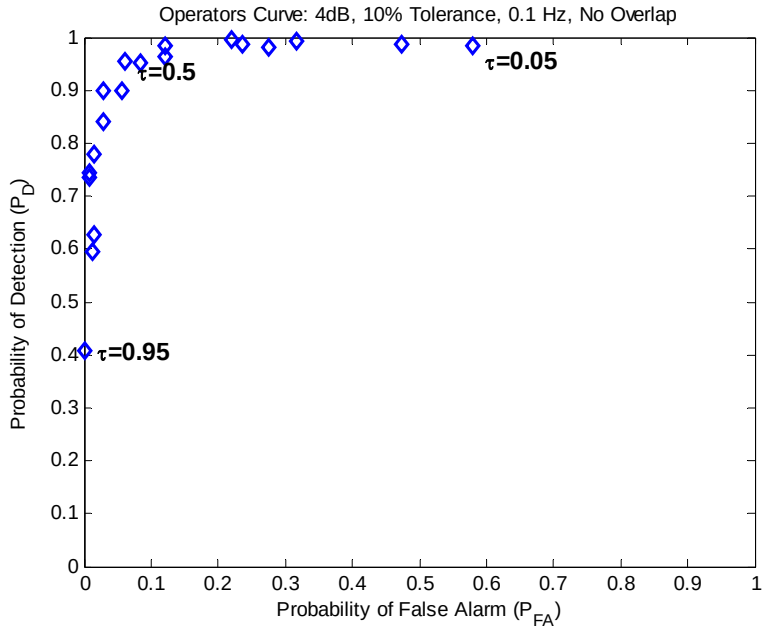


Figure 88. Operating Characteristic Curve, QPSK + AWGN, SNR = 4dB, Difference filter implementation.

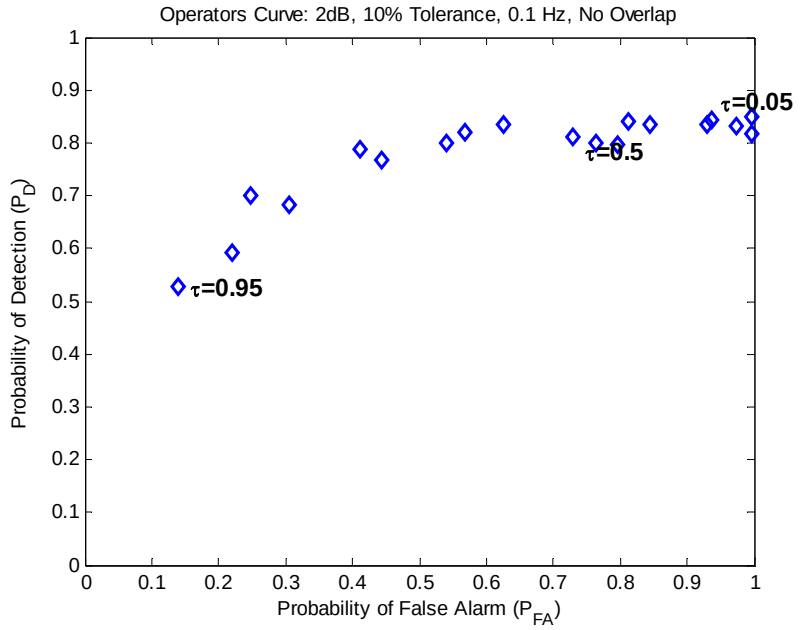


Figure 89. Operating Characteristic Curve, QPSK + AWGN, SNR = 2dB, Difference filter implementation.

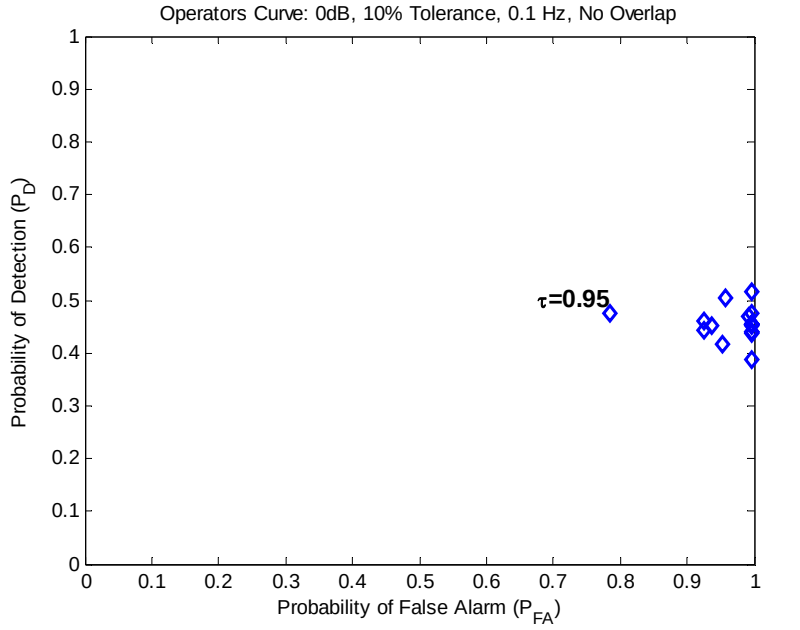


Figure 90. Operating Characteristic Curve, QPSK + AWGN, SNR = 0dB, Difference filter implementation.

3. OPERATING CURVES OBTAINED FOR THE SG FILTER IMPLEMENTATION OF THE DIFFERENTIATION STEP

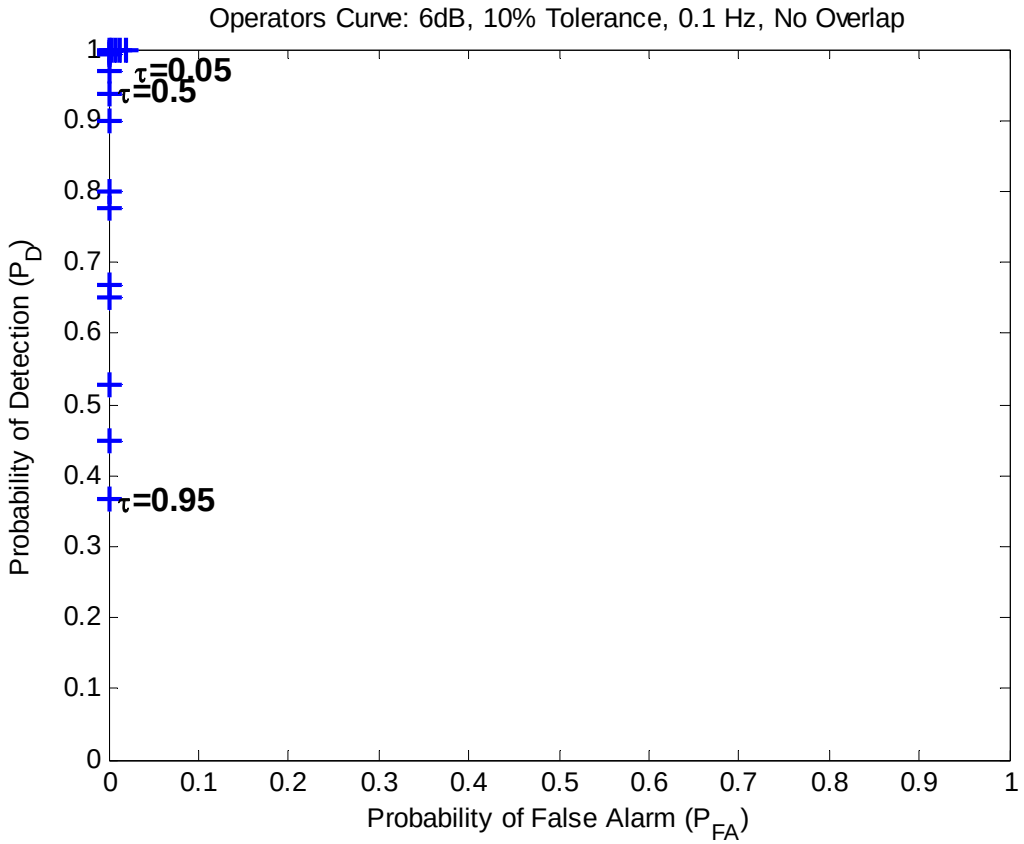


Figure 91. Operating Characteristic Curve, QPSK + AWGN, SNR = 6dB, SG filter implementation.

LIST OF REFERENCES

[1] W. Stallings, *Data & Computer Communications*, 2nd ed. Upper Saddle River, New Jersey 07458, Prentice-Hall, Inc., 1996.

[2] L. W. I. Couch, *Digital and Analog Communication Systems*, 2nd ed. New York, New York: Macmillan Publishing Company, 1987, pp. 307–320.

[3] H. F. Overdyk, *Detection And Estimation of Frequency Hopping Signals Using Wavelet Transforms*, MSEE Thesis, Naval Postgraduate School, Sep. 1997.

[4] Y. Cheng, *Detection of Frequency Hopping Signals Timing Information Using The Temporal Correlation Function*, MSEE Thesis, Naval Postgraduate School, Sep. 2008.

[5] The MathWorks. MATLAB signal processing toolbox™ documentation: SGOLAY function. [Online] http://www.mathworks.com/access/helpdesk/help/toolbox/signal/index.html?/access/helpdesk/help/toolbox/signal/sgolay.html&http://www.mathworks.com/access/helpdesk/help/toolbox/signal/f9-131178.html, Aug. 2009.

[6] S. E. Umbaugh, *Computer Vision and Image Processing: A Practical Approach using CVIPtools.* New Jersey, Prentice Hall, Inc., 1998.

[7] K. R. Castleman, "Digital image processing," in Anonymous Englewood Cliffs, New Jersey, 07632: Prentice Hall, 1996, pp. 499–501.

[8] E. R. Dougherty and R. A. Lotufo, *Hands-on Morphological Image Processing.* Bellingham, Washington: SPIE, 2003, pp. 268.

[9] The MathWorks. MATLAB signal processing toolbox™ documentation. [Online], http://www.mathworks.com/access/helpdesk/help/techdoc/index.html?/access/helpdesk/help/techdoc/matlab.html&http://www.mathworks.com/support/product/product.html?product=SV , Aug. 2009.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Chairman, Code EC
   Department of Electrical and Computer Engineering
   Naval Postgraduate School
   Monterey, California

4. Prof. Monique P. Fargues, Code EC/Fa
   Naval Postgraduate School
   Monterey, California

5. Prof. Roberto Cristi, Code EC/Cx
   Naval Postgraduate School
   Monterey, California